

NNN		NNN	MMM	MMM	LLL
NNN		NNN	MMM	MMM	LLL
NNN		NNN	MMM	MMM	LLL
NNN		NNN	MMMMMM	MMMMMM	LLL
NNN		NNN	MMMMMM	MMMMMM	LLL
NNN		NNN	MMMMMM	MMMMMM	LLL
NNNNNN		NNN	MMM	MMM	LLL
NNNNNN		NNN	MMM	MMM	LLL
NNNNNN		NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNNNNN	NNN	MMM	MMM	LLL
NNN	NNNNNN	NNN	MMM	MMM	LLL
NNN	NNNNNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLLLLLLLLLLLLLLL
NNN	NNN	NNN	MMM	MMM	LLLLLLLLLLLLLLLL
NNN	NNN	NNN	MMM	MMM	LLLLLLLLLLLLLLLL

_S

Ps

NP

NP

SG

SO

NP

PA

_L

```

NN      NN  MM      MM  LL      LL      IIIIII  SSSSSSSS  PPPPPPPP  RRRRRRRR  MM      MM
NN      NN  MM      MM  LL      LL      IIIIII  SSSSSSSS  PPPPPPPP  RRRRRRRR  MM      MM
NN      NN  MMMM    MMMM LL      LL      II      SS      PP      PP  RR      RR  MMMM    MMMM
NN      NN  MMMM    MMMM LL      LL      II      SS      PP      PP  RR      RR  MMMM    MMMM
NNNN    NN  MM      MM  LL      LL      II      SS      PP      PP  RR      RR  MM      MM
NNNN    NN  MM      MM  LL      LL      II      SS      PP      PP  RR      RR  MM      MM
NN      NN  NN      NN  LL      LL      II      SSSSSS  PPPPPPPP  RRRRRRRR  MM      MM
NN      NN  NN      NN  LL      LL      II      SSSSSS  PPPPPPPP  RRRRRRRR  MM      MM
NN      NNNN  MM      MM  LL      LL      II      SS      PP      RR      MM      MM
NN      NNNN  MM      MM  LL      LL      II      SS      PP      RR      MM      MM
NN      NN      MM      MM  LL      LL      II      SS      PP      RR      MM      MM
NN      NN      MM      MM  LL      LL      II      SS      PP      RR      MM      MM
NN      NN      MM      MM  LL      LL      IIIIII  SSSSSSSS  PP      RR      MM      MM
NN      NN      MM      MM  LLLLLLLLLL  LLLLLLLLLL  IIIIII  SSSSSSSS  PP      RR      MM      MM
NN      NN      MM      MM  LLLLLLLLLL  LLLLLLLLLL  IIIIII  SSSSSSSS  PP      RR      MM      MM

```

```

LL      IIIIII  SSSSSSSS
LL      IIIIII  SSSSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SSSSSS
LL      II      SSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LLLLLLLLLL  IIIIII  SSSSSSSS
LLLLLLLLLL  IIIIII  SSSSSSSS

```

```
0001 0 %TITLE 'NML special parameter handling routines'
0002 0 MODULE NML$LISPRM (
0003 0     LANGUAGE (BLISS32),
0004 0     ADDRESSING_MODE (NONEXTERNAL=GENERAL),
0005 0     ADDRESSING_MODE (EXTERNAL=GENERAL),
0006 0     IDENT = 'V04-000'
0007 0 ) =
0008 1 BEGIN
0009 1
0010 1 *****
0011 1 *
0012 1 *  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0013 1 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0014 1 *  ALL RIGHTS RESERVED.
0015 1 *
0016 1 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0017 1 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0018 1 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0019 1 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0020 1 *  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0021 1 *  TRANSFERRED.
0022 1 *
0023 1 *  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0024 1 *  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0025 1 *  CORPORATION.
0026 1 *
0027 1 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0028 1 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0029 1 *
0030 1 *
0031 1 *****
0032 1
0033 1
0034 1 ++
0035 1 FACILITY: DECnet-VAX V2.0 Network Management Listener
0036 1
0037 1 ABSTRACT:
0038 1
0039 1     This module contains action routines to handle changing and
0040 1     displaying of permanent data base entity parameters.
0041 1
0042 1 ENVIRONMENT: VAX/VMS Operating System
0043 1
0044 1 AUTHOR: Distributed Systems Software Engineering
0045 1
0046 1 CREATION DATE: 23-JAN-1980
0047 1
0048 1 MODIFIED BY:
0049 1
0050 1     V03-008 MKP0009      Kathy Perko      2-Aug-1984
0051 1     Fix DEFINE EXEC ADDR n so that, if n doesn't include an area
0052 1     number, area 1 is used.
0053 1
0054 1     V03-007 MKP0008      Kathy Perko      20-April-1984
0055 1     Fix DEF NODE nnn ADDR yyy so that, if the address is a duplicate
0056 1     of the executor's, the error message indicates "executor"
0057 1     instead of "remote node".
```


58	0058	1	
59	0059	1	
60	0060	1	V03-006 MKP0007 Kathy Perko 18-April-1984
61	0061	1	Fix DEF EXEC NAME or ADDRESS so that exec id globals
62	0062	1	are updated.
63	0063	1	
64	0064	1	V03-005 MKP0006 Kathy Perko 29-Jan-1984
65	0065	1	If NCP is a V3.0.0, mask area in node numbers.
66	0066	1	
67	0067	1	V03-004 MKP0005 Kathy Perko 4-Aug-1983
68	0068	1	Change routines to manipulate permanent database record
69	0069	1	fields to be transparent to ISAM keys at the beginning of
70	0070	1	the records. Also, redo checking on node ids for the new
71	0071	1	node database format.
72	0072	1	
73	0073	1	V03-003 MKP0004 Kathy Perko 29-July-1983
74	0074	1	Redo NML\$LISNODEID routine to return only the node id if
75	0075	1	the PSTs datatype is NMASH_PTY_CM1.
76	0076	1	
77	0077	1	V03-002 MKP0003 Kathy Perko 13-July-1982
78	0078	1	Fix NML\$LISPARAM to add parameter lengths correctly.
79	0079	1	Fix list routines for channels and set passwords.
80	0080	1	
81	0081	1	V03-001 MKP0002 Kathy Perko 16-June-1982
82	0082	1	Add new list routines for range and circuit owner paramters.
83	0083	1	
84	0084	1	V02-001 MKP0001 Kathy Perko 2-April-1982
85	0085	1	Add changes for X-25 Protocol Networks and DTE, and
86	0086	1	for X-25 Server Modules.
87	0087	1	
88	0088	1	V02-001 MKP001 Kathy Perko 24-July-1981
89	0089	1	Delete NML call to map VMS line to DNA line name and
90	0090	1	vice versa.
91	0091	1	--

```
93 0092 1 %SBTTL 'Declarations'
94 0093 1
95 0094 1
96 0095 1 TABLE OF CONTENTS:
97 0096 1
98 0097 1
99 0098 1 FORWARD ROUTINE
100 0099 1 NML$NISNMLVER,
101 0100 1 NML$NISLOONAM,
102 0101 1 NML$NISNODEID,
103 0102 1 NML$NISPARAM,
104 0103 1 NML$NISPASSWORD,
105 0104 1 NML$NISPWSET,
106 0105 1 NML$NISRANGE,
107 0106 1 NML$NISOWNER,
108 0107 1 NML$DEFPARAM,
109 0108 1 NML$DEFLINLT,
110 0109 1 NML$DEFLINTRI,
111 0110 1 NML$DEF_NODE_ADDR,
112 0111 1 NML$DEF_EXEC_ID,
113 0112 1 NML_FIND_DUPLICATE_NODE,
114 0113 1 NML$DEFNODNLI,
115 0114 1 NML$DEFOBJNUM,
116 0115 1 NML$PURPARAM,
117 0116 1 NML$PURNODNNA;
118 0117 1
119 0118 1
120 0119 1 INCLUDE FILES:
121 0120 1
122 0121 1
123 0122 1 LIBRARY 'LIB$:NMLLIB.L32';
124 0123 1 LIBRARY 'SHRLIB$:NMLIBRY.L32';
125 0124 1 LIBRARY 'SYSSLIBRARY:STARLET.L32';
126 0125 1
127 0126 1
128 0127 1 OWN STORAGE:
129 0128 1
130 0129 1
131 0130 1
132 0131 1 Parameter buffer and descriptor for use in handling volatile data base
133 0132 1 data.
134 0133 1
135 0134 1 OWN
136 0135 1 nml$t_prmbuffer : VECTOR [256, BYTE];
137 0136 1 BIND
138 0137 1 nml$q_prmdsc = UPLIT (256, nml$t_prmbuffer) : DESCRIPTOR;
139 0138 1
140 0139 1 Entity buffer and descriptor.
141 0140 1
142 0141 1 OWN
143 0142 1 nml$t_entbuffer : BBLOCK [nml$k_entbuflen],
144 0143 1 nml$q_entbfdsc : VECTOR [2];
145 0144 1
146 0145 1
147 0146 1 EXTERNAL REFERENCES:
148 0147 1
149 0148 1
```

```
: 150      0149 1 $NML_EXTDEF;
: 151      0150 1
: 152      0151 1 EXTERNAL LITERAL
: 153      0152 1     nml$_recbfov,
: 154      0153 1     nml$_recdelet;
: 155      0154 1
: 156      0155 1 EXTERNAL
: 157      0156 1     nml$gw_perm_exec_addr : BBLOCK [2],
: 158      0157 1     nml$gb_ncp_version,
: 159      0158 1     nml$gq_perm_exec_name_dsc : VECTOR [2],
: 160      0159 1     nml$gq_proprvmsk : BBLOCK [8];
: 161      0160 1
: 162      0161 1 EXTERNAL ROUTINE
: 163      0162 1     nma$deletefld,
: 164      0163 1     nma$insertfld,
: 165      0164 1     nma$matchrec,
: 166      0165 1     nma$searchfld,
: 167      0166 1     nml$addmsgprm,
: 168      0167 1     nml$bld_reply,
: 169      0168 1     nml$delete_node_rec,
: 170      0169 1     nml$getexeadr,
: 171      0170 1     nml$getnodnam,
: 172      0171 1     nml$getrecowner,
: 173      0172 1     nml$read_loopnode,
: 174      0173 1     nml$readrecord,
: 175      0174 1     nml$send;
: 176      0175 1
```



```
178 0176 1 %SBTTL 'NML$LISNMLVER Get NML version number'
179 0177 1 GLOBAL ROUTINE NML$LISNMLVER (SEM_TABLE, BUFDSC, MSGSIZE, DUMDSC) =
180 0178 1
181 0179 1 ++
182 0180 1 FUNCTIONAL DESCRIPTION:
183 0181 1
184 0182 1 This routine moves the network management version number into
185 0183 1 the output message as a coded multiple parameter.
186 0184 1
187 0185 1 FORMAL PARAMETERS:
188 0186 1
189 0187 1 SEM_TABLE Parameter semantic table entry address.
190 0188 1 BUFDSC Output message buffer descriptor.
191 0189 1 MSGSIZE Address of current output message size.
192 0190 1 DUMDSC Not used.
193 0191 1
194 0192 1 IMPLICIT INPUTS:
195 0193 1
196 0194 1 It is assumed that the permanent data base file is already open.
197 0195 1
198 0196 1 IMPLICIT OUTPUTS:
199 0197 1
200 0198 1 Parameter is added to output message buffer.
201 0199 1
202 0200 1 ROUTINE VALUE:
203 0201 1 COMPLETION CODES:
204 0202 1
205 0203 1 Always returns success (NML$STS_SUC).
206 0204 1
207 0205 1 SIDE EFFECTS:
208 0206 1
209 0207 1 NONE
210 0208 1
211 0209 1 --
212 0210 1
213 0211 2 BEGIN
214 0212 2
215 0213 2 MAP
216 0214 2 SEM_TABLE : REF BBLOCK;
217 0215 2
218 0216 2 LOCAL
219 0217 2 BUFFER : VECTOR [6, BYTE],
220 0218 2 PTR;
221 0219 2
222 0220 2 PTR = CH$PTR (BUFFER); ! Get pointer to output buffer
223 0221 2
224 0222 2
225 0223 2 Add version numbers preceded by data type.
226 0224 2
227 0225 2 CH$WCHAR_A (1, PTR);
228 0226 2 CH$WCHAR_A (NML$K_VERSION, PTR);
229 0227 2 CH$WCHAR_A (1, PTR);
230 0228 2 CH$WCHAR_A (NML$K_DEC_ECO, PTR);
231 0229 2 CH$WCHAR_A (1, PTR);
232 0230 2 CH$WCHAR_A (NML$K_USER_ECO, PTR);
233 0231 2
234 0232 2
```

```
: 235      0233 2 ! Add coded multiple version parameter to message.
: 236      0234 2 !
: 237      0235 2      NML$ADDMSGPRM (.BUFDSC,
: 238      0236 2          .MSGSIZE,
: 239      0237 2          .SEM_TABLE [PST$W_DATAID],
: 240      0238 2          .SEM_TABLE [PST$B_DATATYPE] OR 3,
: 241      0239 2          6,
: 242      0240 2          BUFFER);
: 243      0241 2
: 244      0242 2      RETURN NML$_STS_SUC
: 245      0243 2
: 246      0244 1      END;
```

! End of NML\$LISNMLVER

```
.TITLE NML$LISPRM NML special parameter handling routi
nes
.IDENT \V04-000\
```

```
.PSECT $SPLITS,NOWRT,NOEXE,2
```

```
00000100, 00000 P.AAA: .LONG 256
00000000, 00004 .ADDRESS NML$T_PRMBUFFER
```

```
.PSECT $OWNS,NOEXE,2
```

```
00000 NML$T_PRMBUFFER:
      .BLKB 256
00100 NML$T_ENTBUFFER:
      .BLKB 64
00140 NML$Q_ENTBFDSC:
      .BLKB 8
```

```
NML$Q_PRMDSC= P.AAA
      .EXTRN NML$GB_EVTSRCTYP
      .EXTRN NML$GQ_EVTSRCDS
      .EXTRN NML$GW_EVTCLASS
      .EXTRN NML$GB_EVTMSKTYP
      .EXTRN NML$GQ_EVTMSKDS
      .EXTRN NML$GW_EVTSNKADR
      .EXTRN NML$GW_ACP_CHAN
      .EXTRN NML$GL_LOGMASK, NML$GQ_ENTSTRDSC
      .EXTRN NML$AB_QIOBUFFER
      .EXTRN NML$GQ_QIOBFDSC
      .EXTRN NML$AB_EXEBUFFER
      .EXTRN NML$GL_EXEDATPTR
      .EXTRN NML$GQ_EXEDATDSC
      .EXTRN NML$GQ_EXEBFDSC
      .EXTRN NML$AB_RCVBUFFER
      .EXTRN NML$GQ_RCVBFDSC
      .EXTRN NML$AB_SNDBUFFER
      .EXTRN NML$GQ_SNDBFDSC
      .EXTRN NML$GL_RCVDATLEN
      .EXTRN NML$AB_CPTABLE, NML$AB_MSGBLOCK
      .EXTRN NML$AB_ENTITY_ID
      .EXTRN NML$AB_QUALIFIER_ID
      .EXTRN NML$AB_ENTITYDATA
      .EXTRN NML$AB_NML_NMV, NML$AB_PRMSEM
```



```

                                0000 00000
                                08 C2 00002
5E                               6E 9E 00005
50                               8F D0 00008
80 00010401                     01 B0 0000F
80                               5E D0 00012
                                06 5D 00014
                                50 04 AC D0 00016
                                51 03 A0 9A 0001A
7E                               03 C9 0001E
                                7E 60 3C 00022
                                7E 08 AC 7D 00025
00000000G 00                   06 FB 00029
50                               01 D0 00030
                                04 00 00033
```

; Routine Size: 52 bytes, Routine Base: \$CODE\$ + 0000

```
.EXTRN NML$AB_RECBUF, NML$AL_ENTINF TAB
.EXTRN NML$AL_PERMINF TAB
.EXTRN NML$AW_PRM_DES, NML$GB_CMD_VER
.EXTRN NML$GB_ENTITY_CODE
.EXTRN NML$GB_ENTITY_FORMAT
.EXTRN NML$GL_QUALIFIER_PST
.EXTRN NML$GB_QUALIFIER_FORMAT
.EXTRN NML$GB_FUNCTION
.EXTRN NML$GB_INFO, NML$GB_OPTIONS
.EXTRN NML$GL_PRCODE, NML$GL_PRS_FLGS
.EXTRN NML$GL_NML_ENTITY
.EXTRN NML$GO_NETNAMDSC
.EXTRN NML$GO_RECBF DSC
.EXTRN NML$GW_PRMDESCNT
.EXTRN NML$ RECBFOVF, NML$ RECDELETE
.EXTRN NML$GW_PERM_EXEC_ADDR
.EXTRN NML$GB_NCP_VERSION
.EXTRN NML$GO_PERM_EXEC_NAME_DSC
.EXTRN NML$GO_PROPRVMSK
.EXTRN NML$DELETEFLD, NML$INSERTFLD
.EXTRN NML$MATCHREC, NML$SEARCHFLD
.EXTRN NML$ADDMSGPRM, NML$BLD_REPLY
.EXTRN NML$DELETE_NODE_REC
.EXTRN NML$GETEXEADR, NML$GETNODNAM
.EXTRN NML$GETREOWNER
.EXTRN NML$READ_LOOPNODE
.EXTRN NML$READRECORD, NML$SEND
```

.PSECT \$CODE\$,NOWRT,2

```
.ENTRY NML$LISNMLVER, Save nothing : 0177
SUBL2 #8, SP :
MOVAB BUFFER, PTR : 0220
MOVL #66561, (PTR)+ : 0225
MOVW #1, (PTR)+ : 0229
PUSHL SP : 0235
PUSHL #6 :
MOVL SEM_TABLE, R0 : 0238
MOVZBL 3(R0), R1 :
BISL3 #3, R1, -(SP) :
MOVZWL (R0), -(SP) : 0237
MOVQ BUFDSC, -(SP) : 0235
CALLS #6, NML$ADDMSGPRM :
MOVL #1, R0 : 0242
RET : 0244
```

```
248 0245 1 %SBTTL 'NML$LISLOONAM Get loop node name'
249 0246 1 GLOBAL ROUTINE NML$LISLOONAM (SEM_LIST, BUFDSC, MSGSIZE, DATDSC)=
250 0247 1
251 0248 1 ++
252 0249 1 FUNCTIONAL DESCRIPTION:
253 0250 1
254 0251 1 This routine returns the loopback node name for a line.
255 0252 1
256 0253 1 FORMAL PARAMETERS:
257 0254 1
258 0255 1 SEM_LIST Parameter semantic table entry address.
259 0256 1 BUFDSC Output message buffer descriptor address.
260 0257 1 MSGSIZE Address of current output message size.
261 0258 1 DATDSC Data buffer descriptor address.
262 0259 1
263 0260 1 IMPLICIT INPUTS:
264 0261 1
265 0262 1 It is assumed that the permanent data base file is already open.
266 0263 1
267 0264 1 ROUTINE VALUE:
268 0265 1 COMPLETION CODES:
269 0266 1
270 0267 1 Always returns success (NML$STS_SUC).
271 0268 1
272 0269 1 SIDE EFFECTS:
273 0270 1
274 0271 1 NONE
275 0272 1
276 0273 1 --
277 0274 1
278 0275 2 BEGIN
279 0276 2
280 0277 2 MAP
281 0278 2 sem_list : REF BBLOCK;
282 0279 2
283 0280 2 LOCAL
284 0281 2 circuit_dsc : VECTOR [2],
285 0282 2 node_dsc : VECTOR [2],
286 0283 2 node_rec_buf: BBLOCK [nm($k_recbflen), ! Buffer for node data
287 0284 2 node_rec_dsc: VECTOR [2], ! Descriptor of node data buffer
288 0285 2 node_rec_data: VECTOR [2], ! Descriptor of data in node
289 0286 2 ! data buffer.
290 0287 2 status;
291 0288 2
292 0289 2
293 0290 2 !
294 0291 2 ! Get the circuit ID from the circuit's permanent database record.
295 0292 2 ! If this fails, it's a bug.
296 0293 2 !
297 0294 2 circuit_dsc [0] = 0;
298 0295 2 circuit_dsc [1] = 0;
299 0296 2 IF NOT nma$searchfld (.datdsc,
300 0297 2 nml$c_key_cir,
301 0298 2 circuit_dsc [0],
302 0299 2 circuit_dsc [1]) THEN
303 0300 2 RETURN nml$sts_mpr;
304 0301 2 node_rec_dsc [0] = nml$k_recbflen;
```

```
0305 2 node_rec_dsc [1] = node_rec_buf;
0306 2 node_rec_data [1] = node_rec_buf;
0307 2
0308 2 Call routine to read through the known loopnodes in the node permanent
0309 2 database, looking for loopnode on the circuit being listed.
0310 2
0311 2 IF nml$read_loopnode (circuit_dsc,
0312 2 node_rec_dsc,
0313 2 node_rec_data) THEN
0314 2
0315 2 BEGIN
0316 2 node_dsc [0] = 0;
0317 2 node_dsc [1] = 0;
0318 2 IF nml$searchfld (node_rec_data,
0319 2 nml$pcno nna,
0320 2 node_dsc [0],
0321 2 node_dsc [1]) THEN
0322 2
0323 2 nml$addmsgprm (.bufdsc,
0324 2 .msgsize,
0325 2 .sem_list [pst$w_dataid],
0326 2 .sem_list [pst$b_datatype],
0327 2 .node_dsc [0],
0328 2 .node_dsc [1]);
0329 2
0330 2 END;
0331 2 RETURN nml$_sts_suc
0332 2 END;
```

! End of NML\$LISLOONAM

52	00000000G	00	9E	00002	.ENTRY	NML\$LISLOONAM, Save R2	0246
5E	FBE0	CE	9E	00009	MOVAB	NML\$SEARCHFLD, R2	
	F8	AD	7C	0000E	MOVAB	-1056(SP), SP	
	FC	AD	9F	00011	CLRQ	CIRCUIT_DSC	0294
	F8	AD	9F	00014	PUSHAB	CIRCUIT_DSC+4	0299
7E	10	AD	04	CE 00017	PUSHAB	CIRCUIT_DSC	0298
		AC	DD	0001A	MNEGL	#4, -(SP)	0296
62		04	FB	0001D	PUSHL	DATDSC	
04		50	E8	00020	CALLS	#4, NML\$SEARCHFLD	
50		0A	CE	00023	BLBS	R0, 1\$	
			04	00026	MNEGL	#10, R0	0300
08	AE	0400	8F	3C 00027	RET		
0C	AE	10	AE	9E 0002D	MOVZWL	#1024, NODE_REC_DSC	0301
04	AE	10	AE	9E 00032	MOVAB	NODE_REC_BUF, NODE_REC_DSC+4	0302
			5E	DD 00037	MOVAB	NODE_REC_BUF, NODE_REC_DATA+4	0303
		0C	AE	9F 00039	PUSHL	SP	0308
		F8	AD	9F 0003C	PUSHAB	NODE_REC_DSC	
00000000G	00	03	FB	0003F	PUSHAB	CIRCUIT_DSC	
31		50	E9	00046	CALLS	#3, NML\$READ_LOOPNODE	
		F0	AD	7C 00049	BLBC	R0, 2\$	
		F4	AD	9F 0004C	CLRQ	NODE_DSC	0312
		F0	AD	9F 0004F	PUSHAB	NODE_DSC+4	0317
7E	01F4	8F	3C	00052	PUSHAB	NODE_DSC	0316
	0C	AE	9F	00057	MOVZWL	#500, -(SP)	0314
		04	FB	0005A	PUSHAB	NODE_REC_DATA	
62		50	E9	0005D	CALLS	#4, NML\$SEARCHFLD	
1A					BLBC	R0, 2\$	

NML\$LISPRM
V04-000

NML special parameter handling routines
NML\$LISLOONAM Get loop node name

H 3
16-Sep-1984 00:16:56 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 12:50:09 [NML.SRC]NMLLISPRM.B32;1

Page 10
(4)

7E	F0	AD	7D	00060	MOVQ	NODE DSC, -(SP)	:	0322
50	04	AC	D0	00064	MOVL	SEM LIST, R0	:	0321
7E	03	A0	9A	00068	MOVZBL	3(R0), -(SP)	:	
7E	60	3C	0006C		MOVZWL	(R0), -(SP)	:	0320
7E	08	AC	7D	0006F	MOVQ	BUFDSC, -(SP)	:	0318
00000000G	00	06	FB	00073	CALLS	#6, NML\$ADDMSGPRM	:	
50	01	D0	0007A	2\$:	MOVL	#1, R0	:	0325
		04	0007D		RET		:	0326

; Routine Size: 126 bytes, Routine Base: \$CODE\$ + 0034

```
331 0327 1 %SBTTL 'NML$NISNODEID Get host node id'
332 0328 1 GLOBAL ROUTINE NML$NISNODEID (SEM_LIST, BUFDSC, MSGSIZE, DATDSC)=
333 0329 1
334 0330 1 ++
335 0331 1 FUNCTIONAL DESCRIPTION:
336 0332 1
337 0333 1 This routine gets the host node identification string.
338 0334 1
339 0335 1 FORMAL PARAMETERS:
340 0336 1
341 0337 1 SEM_LIST Parameter semantic table entry address.
342 0338 1 BUFDSC Output message buffer descriptor address.
343 0339 1 MSGSIZE Address of current output message size.
344 0340 1 DATDSC Data buffer descriptor address.
345 0341 1
346 0342 1 IMPLICIT INPUTS:
347 0343 1
348 0344 1 It is assumed that the permanent data base file is already open.
349 0345 1
350 0346 1 IMPLICIT OUTPUTS:
351 0347 1
352 0348 1 NONE
353 0349 1
354 0350 1 ROUTINE VALUE:
355 0351 1 COMPLETION CODES:
356 0352 1
357 0353 1 Always returns success (NML$STS_SUC).
358 0354 1
359 0355 1 SIDE EFFECTS:
360 0356 1
361 0357 1 NONE
362 0358 1
363 0359 1 --
364 0360 1
365 0361 2 BEGIN
366 0362 2
367 0363 2 MAP
368 0364 2 sem_list : REF BBLOCK;
369 0365 2
370 0366 2 OWN
371 0367 2 tmpbuffer : BBLOCK [6];
372 0368 2 BIND
373 0369 2 tmpdsc = UPLIT (6, tmpbuffer) : DESCRIPTOR;
374 0370 2
375 0371 2 LOCAL
376 0372 2 cm_count,
377 0373 2 fldadr,
378 0374 2 fldsize,
379 0375 2 length,
380 0376 2 namdsc : DESCRIPTOR,
381 0377 2 hostadr : WORD,
382 0378 2 ptr,
383 0379 2 reslen;
384 0380 2
385 0381 2 fldadr = 0;
386 0382 2
387 0383 2 IF NOT nma$searchfld (.datdsc,
```

```
388      .sem_list [pst$w_dataid],  
389      fldsize,  
390      fldadr) THEN  
391      RETURN nml$sts_pty;  
392  
393      ptr = nml$st_prmbuffer;  
394  
395      Get the maximum number of fields in the coded multiple: 1 (node address  
396      only) or 2 (node address and node name).  
397  
398      cm_count = .sem_list [pst$b_datatype] AND NOT nma$m_pty_cmu;  
399  
400      hostadr = .(.fldadr)<0,16>;  
401  
402      Add node address field.  
403  
404      CH$WCHAR_A (2, ptr);  
405  
406      If the NCP I'm talking to is speaking NICE V3.0.0 or less, clear the  
407      area number from node numbers in the executor's area.  
408  
409      IF CH$RCHAR (nml$gb_ncp_version) LEQ 3 THEN  
410      BEGIN  
411      MAP  
412      hostadr : BBLOCK [2];  
413  
414      IF .hostadr [nma$v_area] EQL .nml$gw_perm_exec_addr [nma$v_area] THEN  
415      hostadr [nma$v_area] = 0;  
416      END;  
417  
418      ptr = CH$MOVE (2, hostadr, .ptr);  
419      IF .cm_count EQL 2 THEN  
420      BEGIN  
421      nml$getnodnam (.hostadr, tmpdsc, reslen);  
422      namdsc [dsc$w_length] = .reslen;  
423      namdsc [dsc$a_pointer] = tmpbuffer;  
424  
425      Add node name field if the length is not zero.  
426  
427      IF .namdsc [dsc$w_length] NEQU 0 THEN  
428      BEGIN  
429      CH$WCHAR_A (nma$m_pty_asc, ptr);  
430      CH$WCHAR_A (.namdsc [dsc$w_length], ptr);  
431      ptr = CH$MOVE (.namdsc [dsc$w_length],  
432      .namdsc [dsc$a_pointer],  
433      .ptr);  
434      END  
435      ELSE  
436      cm_count = 1;  
437      END;  
438  
439      length = .ptr - nml$st_prmbuffer;  
440      nml$addmsgprm (.bufdsc,  
441      .msgsize,  
442      .sem_list [pst$w_dataid],  
443      nma$m_pty_cmu OR .cm_count,  
444      .length,
```



```

: 445      0441 2      nml$st_prmbuffer);
: 446      0442 2
: 447      0443 2 RETURN nml$_sts_suc
: 448      0444 1 END;
```

! End of NML\$NISNODEID

.PSECT \$SPLITS,NOWRT,NOEXE,2

```
00000006 00008 P.AAB: .LONG 6
00000000 0000C .ADDRESS TMPBUFFER
```

.PSECT \$OWNS,NOEXE,2

```
00148 TMPBUFFER:
      .BLKB 6
```

TMPDSC= P.AAB

.PSECT \$CODE\$,NOWRT,2

			58	00000000'	00	01FC	00000	.ENTRY	NML\$NISNODEID, Save R2,R3,R4,R5,R6,R7,R8	0328
			5E		10	9E	00002	MOVAB	NML\$T_PRMBUFFER, R8	
					7E	C2	00009	SUBL2	#16, SP	
					5E	D4	0000C	CLRL	FLDADR	0381
					5E	DD	0000E	PUSHL	SP	0383
				08	AE	9F	00010	PUSHAB	FLDSIZE	
			56	04	AC	DD	00013	MOVL	SEM_LIST, R6	0384
			7E		66	3C	00017	MOVZWL	(R6), -(SP)	
				10	AC	DD	0001A	PUSHL	DATDSC	0383
		00000000G	00		04	FB	0001D	CALLS	#4, NML\$SEARCHFLD	
			04		50	E8	00024	BLBS	R0, 1\$	
			50		0C	CE	00027	MNEGL	#12, R0	0387
					04	00	0002A	RET		
			53		68	9E	0002B	1\$: MOVAB	NML\$T_PRMBUFFER, PTR	0389
57	03	A6	06		00	EF	0002E	EXTZV	#0, #8, 3(R6), CM_COUNT	0394
			50	00	BE	B0	00034	MOVW	@FLDADR, HOSTADR	0396
			83		02	90	00038	MOVW	#2, (PTR)+	0400
			03	00000000G	00	91	0003B	CMPB	NML\$GB_NCP_VERSION, #3	0405
					15	1A	00042	BGTRU	2\$	
51	00000000G	00	06		02	EF	00044	EXTZV	#2, #6, NML\$GW_PERM_EXEC_ADDR+1, R1	0410
51		50	06		0A	ED	0004D	CMPZV	#10, #6, HOSTADR, RT	
					05	12	00052	BNEQ	2\$	
			50	FC00	8F	AA	00054	BICW2	#64512, HOSTADR	0411
			83		50	B0	00059	MOVW	HOSTADR, (PTR)+	0414
			02		57	D1	0005C	CMPL	CM_COUNT, #2	0415
					35	12	0005F	BNEQ	4\$	
				08	AE	9F	00061	PUSHAB	RESLEN	0417
				00000000'	00	9F	00064	PUSHAB	TMPDSC	
			7E		50	3C	0006A	MOVZWL	HOSTADR, -(SP)	
		00000000G	00		03	FB	0006D	CALLS	#3, NML\$GETNODNAM	
		0C	AE	08	AE	B0	00074	MOVW	RESLEN, NAMDSC	0418
		10	AE	0148	CB	9E	00079	MOVAB	TMPBUFFER, NAMDSC+4	0419
			50	0C	AE	3C	0007F	MOVZWL	NAMDSC, R0	0423
					0E	13	00083	BEQL	3\$	
			83	40	8F	90	00085	MOVW	#64, (PTR)+	0425

NML\$LISPRM
V04-000

NML special parameter handling routines
NML\$NISNODEID Get host node id

L 3
16-Sep-1984 00:16:56
14-Sep-1984 12:50:09

VAX-11 Bliss-32 V4.0-742
[NML.SRC]NMLLISPRM.B32;1

Page 14
(5)

63	10	83	50	90	00089
		BE	50	28	0008C
			03	11	00091
		57	01	D0	00093
		50	68	9E	00096
50		53	50	C3	00099
			8F	BB	0009D
		0101	8F	C9	000A1
7E		57 000000C0	66	3C	000A9
		7E	AC	7D	000AC
		7E	06	FB	000B0
		08	01	D0	000B7
00000000G		00	04	000BA	
		50			

MOVB	R0, (PTR)+
MOVCL	R0, @NAMDS+4, (PTR)
BRB	4\$
MOVL	#1, CM COUNT
MOVAB	NML\$T_PRMBUFFER, R0
SUBL3	R0, PTR, LENGTH
PUSHR	#^M<R0, R8>
BISL3	#192, CM COUNT, -(SP)
MOVZWL	(R6), -(SP)
MOVQ	BUFDSC, -(SP)
CALLS	#6, NML\$ADDMSGPRM
MOVL	#1, R0
RET	

..	0426
..	0429
..	0423
..	0432
..	0435
..	0440
..	0439
..	0438
..	0436
..	0443
..	0444

; Routine Size: 187 bytes, Routine Base: \$CODE\$ + 00B2

```
450 0445 1 %SBTTL 'NML$LISPARAM Get parameter'
451 0446 1 GLOBAL ROUTINE NML$LISPARAM (SEM_LIST, BUFDSC, MSGSIZE, DATDSC)=
452 0447 1
453 0448 1 ++
454 0449 1 FUNCTIONAL DESCRIPTION:
455 0450 1
456 0451 1     This routine returns a parameter.
457 0452 1
458 0453 1 FORMAL PARAMETERS:
459 0454 1
460 0455 1     SEM_LIST      Parameter semantic table entry address.
461 0456 1     BUFDSC        Output message buffer descriptor address.
462 0457 1     MSGSIZE      Address of current output message size.
463 0458 1     DATDSC        QIO buffer descriptor address.
464 0459 1
465 0460 1 IMPLICIT INPUTS:
466 0461 1
467 0462 1     It is assumed that the permanent data base file is already open.
468 0463 1
469 0464 1 IMPLICIT OUTPUTS:
470 0465 1
471 0466 1     The output message buffer contains the coded multiple version number.
472 0467 1
473 0468 1 ROUTINE VALUE:
474 0469 1 COMPLETION CODES:
475 0470 1
476 0471 1     Always returns success (NML$STS_SUC).
477 0472 1
478 0473 1 SIDE EFFECTS:
479 0474 1
480 0475 1     NONE
481 0476 1
482 0477 1 --
483 0478 1
484 0479 2 BEGIN
485 0480 2
486 0481 2 MAP
487 0482 2     SEM_LIST : REF BBLOCK;
488 0483 2
489 0484 2 LOCAL
490 0485 2     DATATYPE : BBLOCK [1],      ! NICE parameter data type.
491 0486 2     FLDADR,
492 0487 2     FLDSIZE;
493 0488 2
494 0489 2     FLDADR = 0;
495 0490 2
496 0491 2     IF NMASSEARCHFLD (.DATDSC,
497 0492 2                     .SEM_LIST [PST$W_DATAID],
498 0493 2                     FLDSIZE,
499 0494 2                     FLDADR)
500 0495 2 THEN
501 0496 2     BEGIN
502 0497 2     DATATYPE = .SEM_LIST [PST$B_DATATYPE];
503 0498 2
504 0499 2     If the parameter is not an ASCII or hex image field, the length
505 0500 2     goes in the datatype byte. Add it here.
506 0501 2
```



```

507 0502 3      IF (NOT .DATATYPE [NMA$V_PTY_ASC]) AND
508 0503 3      (.DATATYPE [NMA$V_PTY_TYP] NEQ NMA$C_PTY_HI) THEN
509 0504 3      DATATYPE = .DATATYPE OR .FLDSIZE;
510 0505 3      NML$ADDMSGPRM (.BUFDSC,
511 0506 3      .MSGSIZE,
512 0507 3      .SEM_LIST [PST$W_DATAID],
513 0508 3      .DATATYPE,
514 0509 3      .FLDSIZE,
515 0510 3      .FLDADR);
516 0511 3      END;
517 0512 3      RETURN NML$_STS_SUC
518 0513 3      END;
519 0514 3      ! End of NML$LISPARAM
```

				0004 00000	.ENTRY NML\$LISPARAM, Save R2	0446
	SE		04	C2 00002	SUBL2 #4, SP	
			7E	D4 00005	CLRL FLDADR	0489
			5E	DD 00007	PUSHL SP	0491
		08	AE	9F 00009	PUSHAB FLDSIZE	
	52	04	AC	D0 0000C	MOVL SEM_LIST, R2	0492
	7E		62	3C 00010	MOVZWL (R2), -(SP)	
		10	AC	DD 00013	PUSHL DATDSC	0491
	00000000G	00	04	FB 00016	CALLS #4, NMA\$SEARCHFLD	
		29	50	E9 0001D	BLBC R0, 2\$	
		50	03	A2 90 00020	MOVB 3(R2), DATATYPE	0497
	OB	50	06	E0 00024	BBS #6, DATATYPE, 1\$	0502
20	50	0F	00	ED 00028	CMPZV #0, #15, DATATYPE, #32	0503
			04	13 0002D	BEQL 1\$	
	50	04	AE	88 0002F	BISB2 FLDSIZE, DATATYPE	0504
			6E	DD 00033	PUSHL FLDADR	0510
		08	AE	DD 00035	PUSHL FLDSIZE	0509
			50	9A 00038	MOVZBL DATATYPE, -(SP)	0508
			62	3C 0003B	MOVZWL (R2), -(SP)	0507
		08	AC	7D 0003E	MOVQ BUFDSC, -(SP)	0505
	00000000G	00	06	FB 00042	CALLS #6, NML\$ADDMSGPRM	
		50	01	D0 00049	MOVL #1, R0	0513
			04	0004C	RET	0514

; Routine Size: 77 bytes, Routine Base: \$CODE\$ + 0160

```
521 0515 1 ZSBTTL 'NML$LISPASSWORD Get parameter'
522 0516 1 GLOBAL ROUTINE NML$LISPASSWORD (SEM_LIST, BUFDSC, MSGSIZE, DATDSC)=
523 0517 1
524 0518 1 ++
525 0519 1 FUNCTIONAL DESCRIPTION:
526 0520 1
527 0521 1 This routine adds a password parameter to the output message if
528 0522 1 the user has the BYPASS privilege.
529 0523 1
530 0524 1 FORMAL PARAMETERS:
531 0525 1
532 0526 1 SEM_LIST Parameter semantic table entry address.
533 0527 1 BUFDSC Output message buffer descriptor address.
534 0528 1 MSGSIZE Address of current output message size.
535 0529 1 DATDSC Address of data buffer descriptor.
536 0530 1
537 0531 1 IMPLICIT INPUTS:
538 0532 1
539 0533 1 It is assumed that the permanent data base file is already open.
540 0534 1
541 0535 1 IMPLICIT OUTPUTS:
542 0536 1
543 0537 1 NONE
544 0538 1
545 0539 1 ROUTINE VALUE:
546 0540 1 COMPLETION CODES:
547 0541 1
548 0542 1 Always returns success (NML$STS_SUC).
549 0543 1
550 0544 1 SIDE EFFECTS:
551 0545 1
552 0546 1 NONE
553 0547 1
554 0548 1 --
555 0549 1
556 0550 2 BEGIN
557 0551 2
558 0552 2 MAP
559 0553 2 SEM_LIST : REF BBLOCK;
560 0554 2
561 0555 2 BIND
562 0556 2 STRDSC = $ASCID ('no access rights') : DESCRIPTOR;
563 0557 2
564 0558 2 LOCAL
565 0559 2 FLDADR,
566 0560 2 FLDSIZE;
567 0561 2
568 0562 2 IF NOT .NML$GQ_PROPRVMSK [PRV$V_BYPASS]
569 0563 2 THEN
570 0564 2 BEGIN
571 0565 2
572 0566 2 User does not have BYPASS privilege so return string to indicate that
573 0567 2 a password is set if one is found.
574 0568 2
575 0569 2
576 0570 2 FLDADR = 0;
577 0571 2 IF NML$SEARCHFLD (.DATDSC,
```

```
578 0572 3 .SEM_LIST [PST$W_DATAID],
579 0573 3 FLD$SIZE,
580 0574 3 FLD$ADR)
581 0575 3 THEN
582 0576 4 BEGIN
583 0577 4
584 0578 4 NML$ADDMSGPRM (.BUF$DSC,
585 0579 4 .MSG$SIZE,
586 0580 4 .SEM_LIST [PST$W_DATAID],
587 0581 4 .SEM_LIST [PST$B_DATA$TYPE],
588 0582 4 .STR$DSC [DSC$W_LENGTH],
589 0583 4 .STR$DSC [DSC$A_POINTER]);
590 0584 4
591 0585 4 RETURN NML$_STS_SUC
592 0586 4
593 0587 4 END;
594 0588 4 END;
595 0589 4
596 0590 4 Call the normal parameter routine.
597 0591 4
598 0592 4 NML$LISPARAM (.SEM_LIST,
599 0593 4 .BUF$DSC,
600 0594 4 .MSG$SIZE,
601 0595 4 .DAT$DSC);
602 0596 4
603 0597 2 RETURN NML$_STS_SUC
604 0598 1 END;

! End of NML$LISPASSWORD
```

```
74 68 67 69 72 20 73 73 65 63 63 61 20 6F 6E 00010 P.AAD: .ASCII \no access rights\
73 0001F
00000010 00020 P.AAC: .LONG 16
00000000 00024 .ADDRESS P.AAD
```

STRDSC= P.AAC

```
0004 00000
08 C2 00002
05 E0 00005
6E D4 0000D
5E DD 0000F
08 AE 9F 00011
52 04 AC D0 00014
7E 62 3C 00018
10 AC DD 0001B
00000000G 00 04 FB 0001E
21 50 E9 00025
00000000' 00 DD 0002B
7E 00000000' 00 3C 0002E
7E 03 A2 9A 00035
7E 62 3C 00039

.PSECT $CODE$,NOWRT,2
.ENTRY NML$LISPASSWORD, Save R2
SUBL2 #8, SP
BBS #5, NML$GQ_PROPRVMSK+3, 1$
CLRL FLD$ADR
PUSHL SP
PUSHAB FLD$SIZE
MOVL SEM_LIST, R2
MOVZWL (R2), -(SP)
PUSHL DAT$DSC
CALLS #4, NML$SEARCHFLD
BLBC R0, 1$
PUSHL STRDSC+4
MOVZWL STRDSC, -(SP)
MOVZBL 3(R2), -(SP)
MOVZWL (R2), -(SP)
```

```
0516
0562
0570
0571
0572
0571
0583
0582
0581
0580
```


NML\$LISPRM
V04-000

NML special parameter handling routines
NML\$LISPASSWORD Get parameter

D 4
16-Sep-1984 00:16:56
14-Sep-1984 12:50:09

VAX-11 Bliss-32 V4.0-742
[NML.SRC]NMLLISPRM.B32;1

Page 19
(7)

00000000G	7E 00	08	AC	7D 0003C	MOVQ	BUFDSC, -(SP)	: 0578
			06	FB 00040	CALLS	#6, NML\$ADDMSGPRM	
			0D	11 00047	BRB	28	: 0585
	7E	0C	AC	7D 00049 18:	MOVQ	MSGSIZE, -(SP)	: 0594
	7E	04	AC	7D 0004D	MOVQ	SEM_LIST, -(SP)	: 0592
FF5D	CF		04	FB C0051	CALLS	#4, NML\$LISPARAM	
	50		01	D0 00056 28:	MOVL	#1, R0	: 0597
			04	00059	RET		: 0598

; Routine Size: 90 bytes, Routine Base: \$CODE\$ + 01BA

```

606 0599 1 XSBTTL 'NML$LISPWSET List password set'
607 0600 1 GLOBAL ROUTINE NML$LISPWSET (SEM_LIST, BUFDSC, MSGSIZE, DATDSC)=
608 0601 1
609 0602 1 ++
610 0603 1 FUNCTIONAL DESCRIPTION:
611 0604 1
612 0605 1 This routine is called while processing a LIST X25-S or X29-S DEST
613 0606 1 command. If a password is set, it adds a password set indicator to
614 0607 1 the NICE response message.
615 0608 1
616 0609 1 FORMAL PARAMETERS:
617 0610 1
618 0611 1 SEM_LIST Parameter semantic table entry address.
619 0612 1 BUFDSC Output message buffer descriptor address.
620 0613 1 MSGSIZE Address of current output message size.
621 0614 1 DATDSC Address of data buffer descriptor.
622 0615 1
623 0616 1 IMPLICIT INPUTS:
624 0617 1
625 0618 1 IMPLICIT OUTPUTS:
626 0619 1
627 0620 1 ROUTINE VALUE:
628 0621 1 COMPLETION CODES:
629 0622 1
630 0623 1 SIDE EFFECTS:
631 0624 1
632 0625 1 --
633 0626 1
634 0627 2 BEGIN
635 0628 2
636 0629 2 MAP
637 0630 2 SEM_LIST : REF BBLOCK;
638 0631 2
639 0632 2 LOCAL
640 0633 2 FLDSIZE,
641 0634 2 FLDADR;
642 0635 2
643 0636 2 IF NMASSEARCHFLD (.DATDSC,
644 0637 2 SEM_LIST [PST$W_DATAID],
645 0638 2 FLDSIZE,
646 0639 2 FLDADR) THEN
647 0640 2 BEGIN
648 0641 2
649 0642 2 Add password to message with a value of 0. This indicates simply that
650 0643 2 the password is defined, without actually returning the password.
651 0644 2
652 0645 2 NML$ADDMSGPRM (.BUFDSC,
653 0646 2 .MSGSIZE,
654 0647 2 SEM_LIST [PST$W_DATAID],
655 0648 2 1,
656 0649 2 1,
657 0650 2 UPLIT (0));
658 0651 2 END;
659 0652 2 RETURN NML$STS_SUC
660 0653 1 END; ! end of NML$LISPWSET
```

```

                                .PSECT $SPLITS,NOWRT,NOEXE,2
                                00000000 00028 P.AAE: .LONG 0

                                .PSECT $CODE$,NOWRT,2
                                .ENTRY NML$LISPWSET, Save nothing
                                SUBL2 #8, SP
                                PUSHL SP
                                PUSHAB FLDSIZE
                                MOVZWL @SEM_LIST, -(SP)
                                PUSHL DATDSC
                                CALLS #4, NML$SEARCHFLD
                                BLBC R0, 1$
                                PUSHAB P.AAE
                                PUSHL #1
                                PUSHL #1
                                MOVZWL @SEM_LIST, -(SP)
                                MOVQ BIJFDC, -(SP)
                                CALLS #6, NML$ADDMSGPRM
                                MOVL #1, R0
                                RET

```

0600
0636
0637
0636
0650
0645
0647
0645
0652
0653

; Routine Size: 56 bytes, Routine Base: \$CODE\$ + 0214

```
662 0654 1 %SBTTL 'NML$LISRANGE List range parameter'
663 0655 1 GLOBAL ROUTINE NML$LISRANGE (SEM_LIST, BUFDSC, MSGSIZE, DATDSC)=
664 0656 1
665 0657 1 ++
666 0658 1 FUNCTIONAL DESCRIPTION:
667 0659 1
668 0660 1 This routine is called to list X25 and X29 Destination subaddresses
669 0661 1 and X25 DTE channels. The destination's subaddresses can be more
670 0662 1 than one range pair, in which case the field length in the permanent
671 0663 1 database is the number of range pairs times 4 (i.e. then length in
672 0664 1 bytes).
673 0665 1
674 0666 1 FORMAL PARAMETERS:
675 0667 1
676 0668 1 SEM_LIST Parameter semantic table entry address.
677 0669 1 BUFDSC Output message buffer descriptor address.
678 0670 1 MSGSIZE Address of current output message size.
679 0671 1 DATDSC Address of data buffer descriptor.
680 0672 1
681 0673 1 --
682 0674 1
683 0675 2 BEGIN
684 0676 2
685 0677 2 MAP
686 0678 2 SEM_LIST : REF BBLOCK;
687 0679 2
688 0680 2 LOCAL
689 0681 2 FLDADR,
690 0682 2 FLDSIZE,
691 0683 2 CM_COUNT,
692 0684 2 LENGTH,
693 0685 2 PTR,
694 0686 2 RANGE_BEGIN,
695 0687 2 RANGE_END;
696 0688 2
697 0689 2 FLDADR = 0;
698 0690 2
699 0691 2 IF NMASSEARCHFLD (.DATDSC,
700 0692 2 SEM_LIST [PST$W_DATAID],
701 0693 2 FLDSIZE,
702 0694 2 FLDADR) THEN
703 0695 2 BEGIN
704 0696 2
705 0697 2 For as many range pairs as are set, add them to the NICE response message
706 0698 2 in the form: Parameter ID, Coded multiple data type, word data type,
707 0699 2 range begin, word data type, range end.
708 0700 2
709 0701 2 WHILE .FLDSIZE GTR 0 DO
710 0702 2 BEGIN
711 0703 2 PTR = NML$T_PMBUFFER;
712 0704 2 CM_COUNT = T;
713 0705 2
714 0706 2 CH$WCHAR A (2, PTR);
715 0707 2 PTR = CH$MOVE (2, (.FLDADR) <0,16>, .PTR);
716 0708 2
717 0709 2 If the range begin = range end, don't include range end.
718 0710 2
```



```
719 0711 4 IF (.FLDADR) <0,16> NEQ (.FLDADR) <16,32> THEN
720 0712 5 BEGIN
721 0713 5 CM_COUNT = .CM_COUNT + 1;
722 0714 5 CH$WCHAR A (2, .PTR);
723 0715 5 PTR = CH$MOVE (2, (.FLDADR) <16,32>, .PTR);
724 0716 4 END;
725 0717 4
726 0718 4 LENGTH = .PTR - NML$T_PRMBUFFER;
727 0719 4 NML$ADDMSGPRM (.BUFDSC,
728 0720 4 .MSGSIZE,
729 0721 4 .SEM_LIST [PST$W_DATAID],
730 0722 4 .SEM_LIST [PST$B_DATATYPE] OR .CM_COUNT,
731 0723 4 .LENGTH,
732 0724 4 NML$T_PRMBUFFER);
733 0725 4
734 0726 4 Increment pointer and length to get next range pair in the
735 0727 4 permanent data base record.
736 0728 4
737 0729 4 FLDADR = .FLDADR + 4;
738 0730 4 FLDSIZE = .FLDSIZE - 4;
739 0731 3 END;
740 0732 2 END;
741 0733 2
742 0734 2 RETURN NML$STS_SUC
743 0735 1 END; ! end of NML$LISRANGE
```

		007C 00000	.ENTRY	NML\$LISRANGE, Save R2,R3,R4,R5,R6	0655
56	00000000	00 9E 00002	MOVAB	NML\$T_PRMBUFFER, R6	
5E		04 C2 00009	SUBL2	#4, SP	
		7E D4 0000C	CLRL	FLDADR	0689
		5E DD 0000E	PUSHL	SP	0691
	08	AE 9F 00010	PUSHAB	FLDSIZE	
7E	04	BC 3C 00013	MOVZWL	@SEM_LIST, -(SP)	0692
	10	AC DD 00017	PUSHL	DATDSC	0691
00000000G	00	04 FB 0001A	CALLS	#4, NMASSEARCHFLD	
56		50 E9 00021	BLBC	R0, 3\$	
53	04	AC D0 00024	MOVL	SEM_LIST, R3	0722
	04	AE D5 00028 1\$:	TSTL	FLDSIZE	0701
		4D 15 0002B	BLEQ	3\$	
52		66 9E 0002D	MOVAB	NML\$T_PRMBUFFER, PTR	0703
54		01 D0 00030	MOVL	#1, CM_COUNT	0704
82		02 90 00033	MOVB	#2, (PTR)+	0706
82	00	BE B0 00036	MOVW	@FLDADR, (PTR)+	0707
50		6E D0 0003A	MOVL	FLDADR, R0	0711
51	02	A0 9E 0003D	MOVAB	2(R0), R1	
51		6E D1 00041	CMPL	FLDADR, R1	
		09 13 00044	BEQL	2\$	
		54 D6 00046	INCL	CM_COUNT	0713
82		02 90 00048	MOVB	#2, (PTR)+	0714
82	02	A0 B0 0004B	MOVW	2(R0), (PTR)+	0715
50		66 9E 0004F 2\$:	MOVAB	NML\$T_PRMBUFFER, R0	0718
55	52	50 C3 00052	SUBL3	R0, PTR, LENGTH	
	0060	8F BB 00056	PUSHR	#*M<R5,R6>	0723

NML\$LISPRM
V04-000

NML special parameter handling routines
NML\$LISRANGE List range parameter

1 4
16-Sep-1984 00:16:56
14-Sep-1984 12:50:09

VAX-11 Bliss-32 V4.0-742
[NML.SRC]NMLLISPRM.B32;1

Page 24
(9)

	50	03	A3	9A	0005A	
7E	50		54	C9	0005E	
	7E	04	BC	3C	00062	
	7E	08	AC	7D	00066	
00000000G	00		06	FB	0006A	
	6E		04	C0	00071	
04	AE		04	C2	00074	
			AE	11	00078	
	50		01	D0	0007A	3\$:
			04	00	0007D	

MOVZBL	3(R3), R0
BISL3	CM COUNT, R0, -(SP)
MOVZWL	@SEM_LIST, -(SP)
MOVQ	BUFDSC, -(SP)
CALLS	#6, NML\$ADDMSGPRM
ADDL2	#4, FLDADR
SUBL2	#4, FLDSIZE
BRB	1\$
MOVL	#1, R0
RET	

:	0722
:	
:	0721
:	0719
:	
:	0729
:	0730
:	0701
:	0734
:	0735

; Routine Size: 126 bytes, Routine Base: \$CODE\$ + 024C

NML
V04

; R

```
745 0736 1 %SBTTL 'NML$LIOWNER Get OWNER parameter'
746 0737 1 GLOBAL ROUTINE NML$LIOWNER (SEM_LIST, BUFDSC, MSGSIZE, DATDSC)=
747 0738 1
748 0739 1 ++
749 0740 1 FUNCTIONAL DESCRIPTION:
750 0741 1 This routine adds the circuit parameter, OWNER, to the NICE
751 0742 1 response message. The owner parameter is saved as a bit value.
752 0743 1 If it's set, the executor owns the circuit. Check to see if
753 0744 1 it's set, and, if so, return the executor node ID.
754 0745 1
755 0746 1 FORMAL PARAMETERS:
756 0747 1
757 0748 1 SEM_LIST Parameter semantic table entry address.
758 0749 1 BUFDSC Output message buffer descriptor address.
759 0750 1 MSGSIZE Address of current output message size.
760 0751 1 DATDSC QIO buffer descriptor address.
761 0752 1
762 0753 1 IMPLICIT INPUTS:
763 0754 1 It is assumed that the permanent data base file is already open.
764 0755 1
765 0756 1 IMPLICIT OUTPUTS:
766 0757 1 The output message buffer contains the coded multiple executor node
767 0758 1 address.
768 0759 1
769 0760 1 ROUTINE VALUE:
770 0761 1 COMPLETION CODES:
771 0762 1 Always returns success (NML$STS_SUC).
772 0763 1
773 0764 1 --
774 0765 1
775 0766 2 BEGIN
776 0767 2
777 0768 2 MAP
778 0769 2 SEM_LIST : REF BBLOCK;
779 0770 2
780 0771 2 BIND EXECUTOR = UPLIT BYTE
781 0772 2 (NMA$M PTY COD+1, NMA$C ENT_NOD, ! Entity type = node
782 0773 2 2, WORD (0)); ! Node address = executor
783 0774 2
784 0775 2 LOCAL
785 0776 2 FLDADR,
786 0777 2 FLDSIZE;
787 0778 2
788 0779 2 FLDADR = 0;
789 0780 2 IF NMA$SEARCHFLD (.DATDSC,
790 0781 2 SEM_LIST [PST$W_DATAID],
791 0782 2 FLDSIZE,
792 0783 2 FLDADR) THEN
793 0784 2 BEGIN
794 0785 2 IF ..FLDADR THEN
795 0786 2 NML$ADDMMSGPRM (.BUFDSC,
796 0787 2 MSGSIZE,
797 0788 2 SEM_LIST [PST$W_DATAID],
798 0789 2 SEM_LIST [PST$B_DATATYPE] OR 2,
799 0790 2 $,
800 0791 2 EXECUTOR);
801 0792 2 END;
```

NML\$LISPRM
V04-000

NML special parameter handling routines
NML\$LIOWNER Get OWNER parameter

K 4
16-Sep-1984 00:16:56
14-Sep-1984 12:50:09

VAX-11 Bliss-32 V4.0-742
[NML.SRC]NMLLISPRM.B32;1

Page 26
(10)

: 802
: 803

0793 2 RETURN NML\$_STS_SUC
0794 1 END;

! End of NML\$LIOWNER

```

                                .PSECT $PLITS$,NOWRT,NOEXE,2
                                02 00 81 0002C P.AAF: .BYTE -127, 0, 2
                                0000 0002F .WORD 0
                                EXECUTOR= P.AAF

                                .PSECT $CODE$,NOWRT,2
                                0004 00000 .ENTRY NML$LIOWNER, Save R2
5E 04 C2 00002 .SUBL2 #4, SP
7E D4 00005 .CLRL FLDADR
08 5E DD 00007 .PUSHL SP
04 AE 9F 00009 .PUSHAB FLDSIZE
52 AC D0 0000C .MOVL SEM_LIST, R2
7E 62 3C 00010 .MOVZWL (R2), -(SP)
10 AC DD 00013 .PUSHL DATDSC
00000000G 00 04 FB 00016 .CALLS #4, NML$SEARCHFLD
22 50 E9 0001D .BLBC R0, 1$
1E 00 BE E9 00020 .BLBC @FLDADR, 1$
00000000 00 00 9F 00024 .PUSHAB EXECUTOR
05 DD 0002A .PUSHL #5
7E 50 03 A2 9A 0002C .MOVZBL 3(R2), R0
5C 02 C9 00030 .BISL3 #2, R0, -(SP)
7E 62 3C 00034 .MOVZWL (R2), -(SP)
08 AC 7D 00037 .MOVQ BUFDSC, -(SP)
00000000G 00 06 FB 0003B .CALLS #6, NML$ADDMSGPRM
50 01 D0 00042 1$: .MOVL #1, R0
04 00045 .RET

```

; Routine Size: 70 bytes, Routine Base: \$CODE\$ + 02CA


```

805 0795 1 %SBTTL 'NML$DEFPARAM Add parameter'
806 0796 1 GLOBAL ROUTINE NML$DEFPARAM (SEM_LIST, BUFSIZE, LENGTH, ADDR, RTNDSC)=
807 0797 1
808 0798 1 !++
809 0799 1 FUNCTIONAL DESCRIPTION:
810 0800 1
811 0801 1 This routine adds a parameter to a permanent data base record.
812 0802 1
813 0803 1 FORMAL PARAMETERS:
814 0804 1
815 0805 1 SEM_LIST Parameter semantic table entry address.
816 0806 1 BUFSIZE Permanent database record maximum size.
817 0807 1 LENGTH Length of parameter to insert in record.
818 0808 1 ADDR Address of parameter to insert in record.
819 0809 1 RTNDSC Permanent database record buffer descriptor address.
820 0810 1
821 0811 1 IMPLICIT INPUTS:
822 0812 1
823 0813 1 It is assumed that the permanent data base file is already open.
824 0814 1
825 0815 1 IMPLICIT OUTPUTS:
826 0816 1
827 0817 1 The parameter is added to the record.
828 0818 1
829 0819 1 ROUTINE VALUE:
830 0820 1 COMPLETION CODES:
831 0821 1
832 0822 1 Always returns success (NML$STS_SUC).
833 0823 1
834 0824 1 SIDE EFFECTS:
835 0825 1
836 0826 1 NONE
837 0827 1
838 0828 1 --
839 0829 1
840 0830 2 BEGIN
841 0831 2
842 0832 2 MAP
843 0833 2 SEM_LIST : REF BBLOCK;
844 0834 2
845 0835 2 IF NOT NMA$INSERTFLD (.BUFSIZE,
846 0836 2 SEM_LIST [PST$W_DATAID],
847 0837 2 LENGTH,
848 0838 2 ADDR,
849 0839 2 RTNDSC)
850 0840 2 THEN
851 0841 2 BEGIN
852 0842 2
853 0843 2 Insert failed.
854 0844 2
855 0845 2 NML$AB_MSGBLOCK [MSB$S_FLAGS] = MSB$M_MSG_FLD; ! Set message text flag
856 0846 2 NML$AB_MSGBLOCK [MSB$S_CODE] = NMA$C_STS_MPR; ! Add error code
857 0847 2 NML$AB_MSGBLOCK [MSB$S_TEXT] = NML$RECBFOVF;
858 0848 2
859 0849 2 RETURN NML$STS_MPR
860 0850 2
861 0851 2 END;
```

```
: 862      0852  2  
: 863      0853  2      RETURN NML$_STS_SUC  
: 864      0854  2  
: 865      0855  1      END;
```

! End of NML\$DEFPARAM

```
                                0004 00000  
52 00000000G 00 9E 00002  
7E          10 AC 7D 00009  
          0C AC DD 0000D  
7E          04 BC 3C 00010  
          08 AC DD 00014  
00000000G 00 05 FB 00017  
13          50 E8 0001E  
62          04 D0 00021  
04 A2          05 8E 00024  
0C A2 00000000G 8F D0 00028  
50          0A CE 00030  
          04 00033  
50          01 D0 00034 1$:  
          04 00037
```

```
.ENTRY NML$DEFPARAM, Save R2  
MOVAB NML$AB_MSGBLOCK, R2  
MOVQ ADDR, -(SP)  
PUSHL LENGTH  
MOVZWL @SEM_LIST, -(SP)  
PUSHL BUFSIZE  
CALLS #5, NML$INSERTFLD  
BLBS R0, 1$  
MOVL #4, NML$AB_MSGBLOCK  
MNEGB #5, NML$AB_MSGBLOCK+4  
MOVL #NML$RECBFOVF, NML$AB_MSGBLOCK+12  
MNEGL #10, R0  
RET  
MOVL #1, R0  
RET
```

```
: 0796  
: 0838  
: 0837  
: 0836  
: 0835  
: 0845  
: 0846  
: 0847  
: 0849  
: 0853  
: 0855
```

; Routine Size: 56 bytes, Routine Base: \$CODE\$ + 0310

```
867 0856 1 %SBTTL 'NML$DEFLINTY Add line type parameter'
868 0857 1 GLOBAL ROUTINE NML$DEFLINTY (SEM_LIST, BUFDSC, LENGTH, ADDR, RTNDSC)=
869 0858 1
870 0859 1 ++
871 0860 1 FUNCTIONAL DESCRIPTION:
872 0861 1
873 0862 1 This routine adds the line type parameter to the permanent data
874 0863 1 base record if the value is valid.
875 0864 1
876 0865 1 FORMAL PARAMETERS:
877 0866 1
878 0867 1 SEM_LIST Parameter semantic table entry address.
879 0868 1 BUFSIZE Permanent database record maximum size.
880 0869 1 LENGTH Length of parameter to insert in record.
881 0870 1 ADDR Address of parameter to insert in record.
882 0871 1 RTNDSC Permanent database record buffer descriptor address.
883 0872 1
884 0873 1 IMPLICIT INPUTS:
885 0874 1
886 0875 1 It is assumed that the permanent data base file is already open.
887 0876 1
888 0877 1 IMPLICIT OUTPUTS:
889 0878 1
890 0879 1 The parameter is added to the record.
891 0880 1
892 0881 1 ROUTINE VALUE:
893 0882 1 COMPLETION CODES:
894 0883 1
895 0884 1 Always returns success (NML$STS_SUC).
896 0885 1
897 0886 1 SIDE EFFECTS:
898 0887 1
899 0888 1 NONE
900 0889 1
901 0890 1 --
902 0891 1
903 0892 2 BEGIN
904 0893 2
905 0894 2 MAP
906 0895 2 SEM_LIST : REF BBLOCK;
907 0896 2
908 0897 2 LOCAL
909 0898 2 FLDADR,
910 0899 2 FLDSIZE,
911 0900 2 STATUS;
912 0901 2
913 0902 2 IF (.ADDR)<0,8> EQL NMA$C_LINTY_POI
914 0903 2 THEN
915 0904 2 BEGIN
916 0905 2
917 0906 2 FLDSIZE = 0;
918 0907 2 IF NMA$SEARCHFLD (.RTNDSC,
919 0908 2 NMA$C_PCLI_TRI,
920 0909 2 FLDSIZE,
921 0910 2 FLDADR)
922 0911 2 THEN
923 0912 2 BEGIN
```

```

: 924      0913  4  !
: 925      0914  4  ! Line has tributary address so it cannot have type=POINT.
: 926      0915  4  !
: 927      0916  4  !
: 928      0917  4      NML$AB_MSGBLOCK [MSB$S_FLAGS] = MSB$M_DET_FLD;
: 929      0918  4      NML$AB_MSGBLOCK [MSB$B_CODE] = NMASC_STS_PVA;
: 930      0919  4      NML$AB_MSGBLOCK [MSB$W_DETAIL] = NMASC_PCLI_LTY;
: 931      0920  4
: 932      0921  4      RETURN NML$_STS_PVA
: 933      0922  4
: 934      0923  4      END;
: 935      0924  4      END;
: 936      0925  4
: 937      0926  4      STATUS = NML$DEFPARAM (.SEM_LIST,
: 938      0927  4          .BUFD$C,
: 939      0928  4          .LENGTH,
: 940      0929  4          .ADDR,
: 941      0930  4          .RTND$C);
: 942      0931  4
: 943      0932  4      RETURN .STATUS
: 944      0933  4
: 945      0934  1      END;
                                ! End of NML$DEFLINLT
```

			0004 00000	.ENTRY	NML\$DEFLINLT, Save R2	0857
	52	00000000G	00 9E 00002	MOVAB	NML\$AB_MSGBLOCK, R2	
	5E		08 C2 00009	SUBL2	#8, SP	
		10	BC 95 0000C	TSTB	@ADDR	0902
			2B 12 0000F	BNEQ	1\$	
		04	AE D4 00011	CLRL	FLDSIZE	0906
			5E DD 00014	PUSHL	SP	0907
		08	AE 9F 00016	PUSHAB	FLDSIZE	
	7E	0474	8F 3C 00019	MOVZWL	#1140, -(SP)	
		14	AC DD 0001E	PUSHL	RTND\$C	
00000000G	00		04 FB 00021	CALLS	#4, NMAS\$SEARCHFLD	
	11		50 E9 00028	BLBC	R0, 1\$	
	62		02 D0 0002B	MOVL	#2, NML\$AB_MSGBLOCK	0917
04	A2		10 8E 0002E	MNEGB	#16, NML\$AB_MSGBLOCK+4	0918
08	A2	0458	8F B0 00032	MOVW	#1112, NML\$AB_MSGBLOCK+8	0919
	50		20 CE 00038	MNEGL	#32, R0	0921
			04 00038	RET		
	7E	10	AC 7D 0003C	MOVQ	ADDR, -(SP)	0929
	7E	08	AC 7D 00040	MOVQ	BUFD\$C, -(SP)	0927
		04	AC DD 00044	PUSHL	SEM_LIST	0926
FF7C	CF		05 FB 00047	CALLS	#5, NML\$DEFPARAM	
			04 0004C	RET		0934

; Routine Size: 77 bytes. Routine Base: \$CODE\$ + 0348


```
947 0935 1 %SBTTL 'NML$DEFLINTRI Add line tributary address parameter'
948 0936 1 GLOBAL ROUTINE NML$DEFLINTRI (SEM_LIST, BUFDSC, LENGTH, ADDR, RTNDSC)=
949 0937 1
950 0938 1 ++
951 0939 1 FUNCTIONAL DESCRIPTION:
952 0940 1
953 0941 1 This routine adds the line tributary address parameter to the
954 0942 1 permanent data base record if it is valid for this line.
955 0943 1
956 0944 1 FORMAL PARAMETERS:
957 0945 1
958 0946 1 SEM_LIST Parameter semantic table entry address.
959 0947 1 BUFSIZE Permanent database record maximum size.
960 0948 1 LENGTH Length of parameter to insert in record.
961 0949 1 ADDR Address of parameter to insert in record.
962 0950 1 RTNDSC Permanent database record buffer descriptor address.
963 0951 1
964 0952 1 IMPLICIT INPUTS:
965 0953 1
966 0954 1 It is assumed that the permanent data base file is already open.
967 0955 1
968 0956 1 IMPLICIT OUTPUTS:
969 0957 1
970 0958 1 The parameter is added to the record.
971 0959 1
972 0960 1 ROUTINE VALUE:
973 0961 1 COMPLETION CODES:
974 0962 1
975 0963 1 Always returns success (NML$STS_SUC).
976 0964 1
977 0965 1 SIDE EFFECTS:
978 0966 1
979 0967 1 NONE
980 0968 1
981 0969 1 --
982 0970 1
983 0971 2 BEGIN
984 0972 2
985 0973 2 MAP
986 0974 2 SEM_LIST : REF BBLOCK;
987 0975 2
988 0976 2 LOCAL
989 0977 2 FLDADR,
990 0978 2 FLDSIZE,
991 0979 2 STATUS;
992 0980 2
993 0981 2 FLDSIZE = 0;
994 0982 2 IF NMA$SEARCHFLD (.RTNDSC,
995 0983 2 NMA$C_PCLI_LTY,
996 0984 2 FLDSIZE,
997 0985 2 FLDADR)
998 0986 2 THEN
999 0987 2 BEGIN
1000 0988 2
1001 0989 2 IF .(FLDADR)<0,8> EQL NMA$C_LINTY_POI
1002 0990 2 THEN
1003 0991 2 BEGIN
```

```
1004 0992 4 1
1005 0993 4 1 Line has type=POINT so no tributary address can be specified.
1006 0994 4 1
1007 0995 4 1
1008 0996 4 1 NML$AB_MSGBLOCK [MSB$SL_FLAGS] = MSB$M_DET_FLD;
1009 0997 4 1 NML$AB_MSGBLOCK [MSB$B_CODE] = NMA$C_STS_PNA;
1010 0998 4 1 NML$AB_MSGBLOCK [MSB$W_DETAIL] = NMA$C_PCLI_TRI;
1011 0999 4 1
1012 1000 4 1 RETURN NML$STS_PNA
1013 1001 4 1
1014 1002 4 1 END;
1015 1003 4 1 END;
1016 1004 4 1
1017 1005 4 1 STATUS = NML$DEFPARAM (.SEM_LIST,
1018 1006 4 1 .BUFD$C,
1019 1007 4 1 .LENGTH,
1020 1008 4 1 .ADDR,
1021 1009 4 1 .RTND$C);
1022 1010 4 1
1023 1011 4 1 RETURN .STATUS
1024 1012 4 1
1025 1013 4 1 END;

! End of NML$DEFLINTRI
```

```
0004 00000
52 00000000G 00 9E 00002
5E 04 AE D4 0000C
08 AE 9F 00011
7E 0458 8F 3C 00014
14 AC DD 00019
00000000G 00 04 FB 0001C
16 50 E9 00023
00 BE 95 00026
11 12 00029
04 62 02 D0 0002B
A2 16 8E 0002E
08 A2 0474 8F B0 00032
50 2C CE 00038
04 04 0003B
7E 10 AC 7D 0003C 1$:
7E 08 AC 7D 00040
04 AC DD 00044
FF2F CF 05 FB 00047
04 0004C
```

```
.ENTRY NML$DEFLINTRI, Save R2
MOVAB NML$AB_MSGBLOCK, R2
SUBL2 #8, SP
CLRL FLDSIZE
PUSHL SP
PUSHAB FLDSIZE
MOVZWL #1112, -(SP)
PUSHL RTND$C
CALLS #4, NMA$SEARCHFLD
BLBC R0, 1$
TSTB @FLDADR
BNEQ 1$
MOVL #2, NML$AB_MSGBLOCK
MNEGB #22, NML$AB_MSGBLOCK+4
MOVW #1140, NML$AB_MSGBLOCK+8
MNEGL #44, R0
RET
MOVQ ADDR, -(SP)
MOVQ BUFD$C, -(SP)
PUSHL SEM_LIST
CALLS #5, NML$DEFPARAM
RET
```

```
0936
0981
0982
0989
0996
0997
0998
1000
1008
1006
1005
1013
```

; Routine Size: 77 bytes, Routine Base: \$CODE\$ + 0395

```
1027 1 XSBTTL 'NML$DEF NODE_ADDR Add node address parameter'
1028 1 GLOBAL ROUTINE NML$DEF_NODE_ADDR (SEM_LIST, BUFDSC, LENGTH, ADDR, RTNDSC)=
1029 1
1030 1 ++
1031 1 FUNCTIONAL DESCRIPTION:
1032 1 This routine checks the node address parameter to make sure
1033 1 it does not already exist in the node permanent database. If it does
1034 1 not, it adds the node address to the permanent data base record.
1035 1 This routine is not used to check for duplicate node names because
1036 1 the node database name key is declared as 'noduplicates', so RMS
1037 1 will do this check for node names when the record is written to
1038 1 the file.
1039 1
1040 1 FORMAL PARAMETERS:
1041 1 SEM_LIST Parameter semantic table entry address.
1042 1 BUFSIZE Permanent database record maximum size.
1043 1 LENGTH Length of parameter to insert in record.
1044 1 ADDR Address of parameter to insert in record.
1045 1 RTNDSC Permanent database record buffer descriptor address.
1046 1
1047 1 IMPLICIT INPUTS:
1048 1 It is assumed that the permanent data base file is already open.
1049 1
1050 1 IMPLICIT OUTPUTS:
1051 1 The parameter is added to the record.
1052 1
1053 1 ROUTINE VALUE:
1054 1 COMPLETION CODES:
1055 1 Returns success (NML$STS_SUC) if the node address is successfully
1056 1 added to the permanent database record.
1057 1 Returns nml$sts_pva if the new address is already defined in the
1058 1 node permanent database.
1059 1
1060 1 SIDE EFFECTS:
1061 1 NONE
1062 1
1063 1 --
1064 1
1065 1 BEGIN
1066 1
1067 1 MAP
1068 1 sem_list : REF BBLOCK,
1069 1 rtndsc : REF DESCRIPTOR;
1070 1
1071 1 LOCAL
1072 1 status;
1073 1
1074 1
1075 1 If there's another node in the permanent database with the new address,
1076 1 return an error message to NCP.
1077 1
1078 1 IF nml_find_duplicate_node (.sem_list, .bufdsc,
1079 1 :length, .addr,
1080 1 :rtndsc) THEN
1081 1 BEGIN
1082 1 nml$ab_msgblock [msb$vdet_fld] = 1;
1083 1 nml$ab_msgblock [msb$code] = nml$sts_pva;
```

```
1084      nml$ab_msgblock [msb$w_detail] = .sem_list [pst$w_dataid];
1085      RETURN nml$sts_pva
1086      END;
1087
1088      !
1089      ! The node address is unique. Add it to the node's permanent database record.
1090      !
1091      status = nml$defparam (.sem_list,
1092                           .bufdsc,
1093                           .length,
1094                           .addr,
1095                           .rtnndsc);
1096      RETURN .status
1097
1098      ! End of NML$DEF_NODE_ADDR
1099      END;
```

			0004 00000	.ENTRY	NML\$DEF NODE ADDR, Save R2	1015
	52	00000000G	00 9E 00002	MOVAB	NML\$AB_MSGBLOCK, R2	
	7E	10	AC 7D 00009	MOVQ	ADDR, -(SP)	1066
	7E	08	AC 7D 0000D	MOVQ	BUFDSC, -(SP)	1065
		04	AC DD 00011	PUSHL	SEM_LIST	
00000000V	00		05 FB 00014	CALLS	#5, NML_FIND_DUPLICATE_NODE	
	10		50 E9 0001B	BLBC	R0, 1\$	
	62		02 88 0001E	BISB2	#2, NML\$AB_MSGBLOCK	1069
04	A2		10 8E 00021	MNEGB	#16, NML\$AB_MSGBLOCK+4	1070
08	A2	04	BC B0 00025	MOVW	@SEM_LIST, NML\$AB_MSGBLOCK+8	1071
	50		20 CE 0002A	MNEGL	#32, R0	1072
			04 0002D	RET		
	7E	10	AC 7D 0002E 1\$:	MOVQ	ADDR, -(SP)	1081
	7E	08	AC 7D 00032	MOVQ	BUFDSC, -(SP)	1079
		04	AC DD 00036	PUSHL	SEM_LIST	1078
FEFO	CF		05 FB 00039	CALLS	#5, NML\$DEFPARAM	
			04 0003E	RET		1086

: Routine Size: 63 bytes, Routine Base: \$CODE\$ + 03E2


```
1101 1087 1 %SBTTL 'NML$DEF_EXEC_ID Add executor name or address parameter'
1102 1088 1 GLOBAL ROUTINE NML$DEF_EXEC_ID (SEM_LIST, BUFDSC, LENGTH, ADDR, RTNDSC)=
1103 1089 1
1104 1090 1 ++
1105 1091 1 FUNCTIONAL DESCRIPTION:
1106 1092 1 This routine is called when processing a DEFINE EXECUTOR command
1107 1093 1 to change the name or address of the executor node. It checks
1108 1094 1 the new name or address parameter to determine if it's already
1109 1095 1 assigned to some other node. If it is, this means the identity
1110 1096 1 of the executor is being changed. Delete the remote entry with
1111 1097 1 that name or address. The new name or address is added to the
1112 1098 1 executor node permanent database record. It is written back
1113 1099 1 to the file later.
1114 1100 1
1115 1101 1 FORMAL PARAMETERS:
1116 1102 1 SEM_LIST Parameter semantic table entry address.
1117 1103 1 BUFSIZE Permanent database record maximum size.
1118 1104 1 LENGTH Length of parameter to insert in record.
1119 1105 1 ADDR Address of parameter to insert in record.
1120 1106 1 RTNDSC Permanent database record buffer descriptor address.
1121 1107 1
1122 1108 1 IMPLICIT INPUTS:
1123 1109 1 It is assumed that the permanent data base file is already open.
1124 1110 1
1125 1111 1 IMPLICIT OUTPUTS:
1126 1112 1 The new executor name or address is added to the record.
1127 1113 1
1128 1114 1 ROUTINE VALUE:
1129 1115 1 COMPLETION CODES:
1130 1116 1 Returns success (NML$STS_SUC) if the node address is successfully
1131 1117 1 added to the permanent database record.
1132 1118 1
1133 1119 1 SIDE EFFECTS:
1134 1120 1 If the new executor name or address is already assigned to some
1135 1121 1 other node in the permanent database, that remote node is deleted from
1136 1122 1 the database.
1137 1123 1
1138 1124 1 --
1139 1125 1
1140 1126 2 BEGIN
1141 1127 2
1142 1128 2 MAP
1143 1129 2 addr : REF BBLOCK [2],
1144 1130 2 sem_list : REF BBLOCK;
1145 1131 2
1146 1132 2 LOCAL
1147 1133 2 status,
1148 1134 2 temp;
1149 1135 2
1150 1136 2 IF nml_find_duplicate_node (.sem_list, .bufdsc,
1151 1137 2 .length, .addr,
1152 1138 2 .rtndsc) THEN
1153 1139 2 BEGIN
1154 1140 2
1155 1141 2 The executor node identity is being changed to that of a node that's
1156 1142 2 already in the database. Delete the remote entry for that node (there
1157 1143 2 are no parameters that it makes sense to carry over in this case)
```

```
1158      ! so the executor can become that node.
1159      !
1160      nml$delete_node_rec (.sem_list [pst$w_dataid],      ! Database key
1161                          length);                        ! Name or address dsc.
1162      nml$ab_msgblock [msb$w_msg_fld] = 1;
1163      nml$ab_msgblock [msb$l_text] = nml$recdelet;
1164      END;
1165
1166      Put the RMS "current record" pointer back to the executor node's
1167      entry.
1168
1169      *****TEMPORARY
1170      nml$gw_perm_exec_addr = 0;
1171      *****
1172      nml$getexeadr (temp);
1173
1174      If the new executor address is 0, leave it that way. If the area number
1175      of the address is 0, then default it to area 1 (this is for DEFINE EXEC
1176      ADDRESS only) so the exec will have a valid area number in the database.
1177
1178      IF .sem_list [pst$w_dataid] EQL nma$c_pcno_add THEN
1179      BEGIN
1180      IF .addr [nma$w_addr] NEQ 0 AND
1181      .addr [nma$w_area] EQL 0 THEN
1182      addr [nma$w_area] = 1;
1183      END;
1184      status = nml$defparam (.sem_list,
1185                          .bufdsc,
1186                          .length,
1187                          .addr,
1188                          .rtndsc);
1189
1190      IF .sem_list [pst$w_dataid] EQL nma$c_pcno_add THEN
1191      nml$gw_perm_exec_addr = (.addr)<0,16>
1192      ELSE
1193      BEGIN
1194      CH$MOVE (.length, .addr, .nml$gq_perm_exec_name_dsc [1]);
1195      nml$gq_perm_exec_name_dsc [0] = .length;
1196      END;
1197      RETURN .status
1198
1199      ! End of NML$DEF_EXEC_ID
```

```
00FC 00000
57 00000000G 00 9E 00002
5E          04 C2 00009
          14 AC DD 0000C
52          10 AC D0 0000F
          52 DD 00013
7E          08 AC 7D 00015
53          04 AC D0 00019
          53 DD 0001D
00000000V 00 05 FB 0001F
```

```
.ENTRY NML$DEF_EXEC_ID, Save R2,R3,R4,R5,R6,R7
MOVAB NML$GW_PERM_EXEC_ADDR, R7
SUBL2 #4, SP
PUSHL RTNDSC
MOVL ADDR, R2
PUSHL R2
MOVQ BUFDSC, -(SP)
MOVL SEM_LIST, R3
PUSHL R3
CALLS #5, NML_FIND_DUPLICATE_NODE
```

```
1088
1138
1137
1136
```

	1F		50	E9	00026	BLBC	R0, 1\$		
		0C	AC	9F	00029	PUSHAB	LENGTH	1146	
	7E		63	3C	0002C	MOVZWL	(R3), -(SP)		
00000000G	00		02	FB	0002F	CALLS	#2, NML\$DELETE_NODE_REC		
00000000G	00		04	88	00036	BISB2	#4, NML\$AB_MSGBLOCK	1148	
00000000G	00	00000000G	8F	D0	0003D	MOVL	#NML\$RECDELET, NML\$AB_MSGBLOCK+12	1149	
			67	D4	00048	CLRL	NML\$GQ_PERM_EXEC_ADDR	1156	
			5E	DD	0004A	PUSHL	SP	1158	
00000000G	00		01	FB	0004C	CALLS	#1, NML\$GETEXEADR		
01F6	8F		63	B1	00053	CMPW	(R3), #502	1164	
			13	12	00058	BNEQ	2\$		
03FF	BF		62	B3	0005A	BITW	(R2), #1023	1166	
			0C	13	0005F	BEQL	2\$		
FC	BF	01	A2	93	00061	BITB	1(R2), #252	1167	
			05	12	00066	BNEQ	2\$		
62		06	0A	F0	00068	INSV	#1, #10, #6, (R2)	1168	
				AC	DD	0006D	PUSHL	RTNDSC	1174
			14	DD	00070	PUSHL	R2	1173	
	7E	08	AC	7D	00072	MOVQ	BUFDSC, -(SP)	1171	
			53	DD	00076	PUSHL	R3	1170	
FE72	CF		05	FB	00078	CALLS	#5, NML\$DEFPARAM		
	56		50	D0	0007D	MOVL	R0, STATUS		
01F6	8F		63	B1	00080	CMPW	(R3), #502	1176	
			05	12	00085	BNEQ	3\$		
	67		62	3C	00087	MOVZWL	(R2), NML\$GQ_PERM_EXEC_ADDR	1177	
			14	11	0008A	BRB	4\$		
	50	00000000G	00	D0	0008C	MOVL	NML\$GQ_PERM_EXEC_NAME_DSC+4, R0	1180	
60	62	0C	AC	28	00093	MOVC3	LENGTH, (R2), (R0)		
	00	0C	AC	D0	00098	MOVL	LENGTH, NML\$GQ_PERM_EXEC_NAME_DSC	1181	
00000000G	50		56	D0	000A0	MOVL	STATUS, R0	1183	
			04	000A3	RET			1185	

; Routine Size: 164 bytes, Routine Base: \$CODE\$ + 0421

```
1201 1186 1 %SBTTL 'NML_FIND_DUPLICATE_NODE Check perm db for node id'
1202 1187 1 ROUTINE NML_FIND_DUPLICATE_NODE (SEM_LIST, BUFDSC,
1203 1188 1 LENGTH, ADDR,
1204 1189 1 RTNDSC)=
1205 1190 1
1206 1191 1
1207 1192 1 ++
1208 1193 1 FUNCTIONAL DESCRIPTION:
1209 1194 1 This routine checks the node name or address parameter to see
1210 1195 1 if it already exists in the node permanent database.
1211 1196 1
1212 1197 1 FORMAL PARAMETERS:
1213 1198 1 SEM_LIST Parameter semantic table entry address.
1214 1199 1 BUFSIZE Permanent database record maximum size.
1215 1200 1 LENGTH Length of parameter to insert in record.
1216 1201 1 ADDR Address of parameter to insert in record.
1217 1202 1 RTNDSC Permanent database record buffer descriptor address.
1218 1203 1
1219 1204 1 IMPLICIT INPUTS:
1220 1205 1 It is assumed that the permanent data base file is already open.
1221 1206 1
1222 1207 1 IMPLICIT OUTPUTS:
1223 1208 1 NML$Q PRMDSC is the descriptor of the duplicate node's record
1224 1209 1 (if there is one) which is used to return the ID of that node
1225 1210 1 in the NICE error message.
1226 1211 1
1227 1212 1 ROUTINE VALUE:
1228 1213 1 COMPLETION CODES:
1229 1214 1 Returns status of node lookup.
1230 1215 1
1231 1216 1 SIDE EFFECTS:
1232 1217 1 None
1233 1218 1
1234 1219 1 --
1235 1220 1
1236 1221 2 BEGIN
1237 1222 2
1238 1223 2 MAP
1239 1224 2 sem_list : REF BBLOCK;
1240 1225 2
1241 1226 2 LOCAL
1242 1227 2 key,
1243 1228 2 node_id_dsc: VECTOR [2],
1244 1229 2 dup_dsc: VECTOR [2],
1245 1230 2 node_type,
1246 1231 2 status;
1247 1232 2
1248 1233 2
1249 1234 2 Look for a node name (or address) that was previously DEFINED in the node's
1250 1235 2 permanent database record.
1251 1236 2
1252 1237 2 node_id_dsc [1] = 0;
1253 1238 2 node_id_dsc [0] = 0;
1254 1239 2 status = nma$searchfld (.rtndsc,
1255 1240 2 .sem_list [p$tw_dataid],
1256 1241 2 node_id_dsc [0],
1257 1242 2 node_id_dsc [1]);
```



```
1258 1243 2
1259 1244 2
1260 1245 2 If there is no previously defined node ID, or the previous ID is different
1261 1246 2 from the new ID in the NICE DEFINE command, then check to see if there's
1262 1247 2 another node with the same name or address in the node permanent database.
1263 1248 2
1264 1249 2 IF NOT .status
1265 1250 2 OR
1266 1251 2 (.status AND
1267 1252 2 CH$NEQ (.node_id_dsc [0], .node_id_dsc [1], .length, .addr)) THEN
1268 1253 2 BEGIN
1269 1254 2 key = .sem_list [pst$w_dataid]; ! Make key a longword.
1270 1255 2 status = nml$readrecord (nma$c_opn_node, ! Node database file ID
1271 1256 2 key, ! Node database key
1272 1257 2 length, ! Address of key value descriptor
1273 1258 2 nml$q_prmdsc, ! Buffer for node record
1274 1259 2 dup_dsc, ! Duplicate node data descriptor
1275 1260 2 node_type); ! Node entity type.
1276 1261 3 IF .status THEN
1277 1262 4 BEGIN
1278 1263 4
1279 1264 4 There is another node with the new name or address DEFINEd.
1280 1265 4 Add duplicate node id to NICE response message parameters. The node
1281 1266 4 ID will be returned in the NICE response to NCP.
1282 1267 4
1283 1268 4 nml$q_entbfdsc [0] = nml$k_entbuflen;
1284 1269 4 nml$q_entbfdsc [1] = nml$t_entbuffer;
1285 1270 4 nml$getrecowner (dup_dsc,
1286 1271 4 .node_type,
1287 1272 4 nml$q_entbfdsc,
1288 1273 4 nml$q_entbfdsc [0]);
1289 1274 4 nml$ab_msgblock [msb$[flags]] = msb$m_entd fld; ! Set entit, descriptor flag
1290 1275 4 nml$ab_msgblock [msb$a_entity] = nml$q_entbfdsc; ! Add entity descriptor pointer
1291 1276 3 END;
1292 1277 3 END
1293 1278 2 ELSE
1294 1279 2 status = nml$sts_cmp;
1295 1280 2 RETURN .status
1296 1281 1 END;

! End of NML_FIND_DUPLICATE_NODE
```

003C 00000 NML_FIND_DUPLICATE_NODE:

55	00000000'	00	9E	00002	WORD	Save R2,R3,R4,R5	1187		
5E		18	C2	00009	MOVAB	NML\$Q_ENTBFDSC, R5			
	10	AE	7C	0000C	SUBL2	#24, SP			
	14	AE	9F	0000F	CLRQ	NODE_ID_DSC	1238		
	14	AE	9F	00012	PUSHAB	NODE_ID_DSC+4	1242		
7E	04	BC	3C	00015	PUSHAB	NODE_ID_DSC	1241		
	14	AC	DD	00019	MOVZWL	@SEM_LIST, -(SP)	1240		
00000000G	00	04	FB	0001C	PUSHL	RTNDSC	1239		
54		50	D0	00023	CALLS	#4, NMASSEARCHFLD			
0C		54	E9	00026	MOVL	R0, STATUS			
OC	AC				BLBC	STATUS, 1\$	1249		
00	14	BE	10	AE	2D	00029	CMPC5	NODE_ID_DSC, @NODE_ID_DSC+4, #0, LENGTH, -	1252

Address	Op Code	Op Name	Comment	Hex
00000000G	00	BC	00031	BEQL
00000000G	00	4F	00033	2\$
00000000G	00	BC	00035	1\$: MOVZWL
00000000G	00	5E	0003A	SEM_LIST, KEY
00000000G	00	OC	0003C	SP
00000000G	00	00	0003F	DUP DSC
00000000G	00	OC	00045	NML\$Q PRMDSC
00000000G	00	14	00048	PUSHAB LENGTH
00000000G	00	7E	0004B	PUSHAB KEY
00000000G	00	06	0004D	CLRL -(SP)
00000000G	00	50	00054	CALLS #6, NML\$READRECORD
00000000G	00	54	00057	RO, STATUS
00000000G	00	8F	0005A	BLBC STATUS, 3\$
00000000G	00	A5	0005E	MOVZBL #64, NML\$Q ENTBF DSC
00000000G	00	55	00063	MOVAB NML\$T_ENTB OFFER, NML\$Q_ENTBF DSC+4
00000000G	00	55	00065	PUSHL R5
00000000G	00	AE	00067	PUSHL R5
00000000G	00	04	0006D	PUSHL NODE TYPE
00000000G	00	10	00074	DUP DSC
00000000G	00	65	0007B	CALLS #4, NML\$GETRE COWNER
00000000G	00	03	00082	MOVL #16, NML\$AB MSGBLOCK
00000000G	00	10	00084	MOVAB NML\$Q_ENTBF DSC, NML\$AB_MSGBLOCK+20
00000000G	00	54	00087	BRB 3\$
00000000G	00	04	0008A	MNEGL #16, STATUS
00000000G	00	54	00087	3\$: MOVL STATUS, RO
00000000G	00	04	0008A	RET

; Routine Size: 139 bytes, Routine Base: \$CODES + 04C5

```
1298 1282 1 %SBTTL 'NML$DEFNODNLI Add loop node line parameter'
1299 1283 1 GLOBAL ROUTINE NML$DEFNODNLI (SEM_LIST, BUFDSC, LENGTH, ADDR, RTNDSC)=
1300 1284 1
1301 1285 1 ++
1302 1286 1 FUNCTIONAL DESCRIPTION:
1303 1287 1
1304 1288 1 This routine adds the loop node line parameter to the permanent
1305 1289 1 data base record if this is a loop node and the circuit id is
1306 1290 1 unique (i.e. there is no other loop node set up on the circuit).
1307 1291 1
1308 1292 1 FORMAL PARAMETERS:
1309 1293 1
1310 1294 1 SEM_LIST      Parameter semantic table entry address.
1311 1295 1 BUFSIZE      Permanent database record maximum size.
1312 1296 1 LENGTH      Length of parameter to insert in record.
1313 1297 1 ADDR        Address of parameter to insert in record.
1314 1298 1 RTNDSC      Permanent database record buffer descriptor address.
1315 1299 1
1316 1300 1 IMPLICIT INPUTS:
1317 1301 1 It is assumed that the permanent data base file is already open.
1318 1302 1
1319 1303 1 IMPLICIT OUTPUTS:
1320 1304 1 The parameter is added to the record.
1321 1305 1
1322 1306 1 ROUTINE VALUE:
1323 1307 1 COMPLETION CODES:
1324 1308 1 Always returns success (NML$STS_SUC).
1325 1309 1
1326 1310 1 SIDE EFFECTS:
1327 1311 1 NONE
1328 1312 1
1329 1313 1 --
1330 1314 1
1331 1315 2 BEGIN
1332 1316 2
1333 1317 2 MAP
1334 1318 2 sem_list : REF BBLOCK;
1335 1319 2
1336 1320 2 LOCAL
1337 1321 2 fldadr,
1338 1322 2 fldsize,
1339 1323 2 circuit_dsc: VECTOR [2], ! Circuit already in node record (if any)
1340 1324 2 node_rec_buf: BBLOCK [nm($k_recbflen)], ! Buffer for node data
1341 1325 2 node_rec_dsc: VECTOR [2], ! Descriptor of node record buffer.
1342 1326 2 node_rec_data: VECTOR [2], ! Descriptor of data in node record buffer.
1343 1327 2 status;
1344 1328 2
1345 1329 2 fldadr = 0;
1346 1330 2 IF nma$searchfld (.rtndsc,
1347 1331 2 nma$c_pcnno_add,
1348 1332 2 fldsize,
1349 1333 2 fldadr) THEN
1350 1334 2 BEGIN
1351 1335 2
1352 1336 2 Node has address so circuit is not allowed. Loopnodes have only one
1353 1337 2 parameter - a circuit ID.
1354 1338 2
```

```
1355 1339 3 nml$ab_msgblock [msb$l_flags] = msb$m_det_fld;
1356 1340 3 nml$ab_msgblock [msb$b_code] = nma$c_sts_pna;
1357 1341 3 nml$ab_msgblock [msb$w_detail] = nma$c_pcho_nli;
1358 1342 3 RETURN nml$sts_pna
1359 1343 3 END;
1360 1344 3
1361 1345 3 circuit_dsc [0] = 0;
1362 1346 3 circuit_dsc [1] = 0;
1363 1347 3 status = nma$searchfld (.rtndsc,
1364 1348 3 nma$c_pcho_nli,
1365 1349 3 circuit_dsc [0],
1366 1350 3 circuit_dsc [1]);
1367 1351 3
1368 1352 3 If the loop node is already set up on the circuit specified in the NICE
1369 1353 3 DEFINE command, I'm done. Otherwise, make sure the circuit isn't already
1370 1354 3 defined for some other loopnode.
1371 1355 3
1372 1356 3 IF NOT .status
1373 1357 3 OR (.status AND CH$NEQ (.circuit_dsc [0], .circuit_dsc [1],
1374 1358 3 .length, .addr)) THEN
1375 1359 3 BEGIN
1376 1360 3 |
1377 1361 3 | Check to make sure there aren't any other loopnodes on the specified
1378 1362 3 | circuit in the node database.
1379 1363 3 |
1380 1364 3 node_rec_dsc [0] = nml$k_recbflen;
1381 1365 3 node_rec_dsc [1] = node_rec_buf;
1382 1366 3 node_rec_data [1] = node_rec_buf;
1383 1367 3 status = nml$read_loopnode (.length, | Address of circuit descriptor
1384 1368 3 node_rec_dsc, | I/O buffer descriptor
1385 1369 3 node_rec_data); | Return node data descriptor
1386 1370 3 IF .status NEQ rms$_eof THEN
1387 1371 3 BEGIN
1388 1372 3 |
1389 1373 3 | Circuit name must be unique for loop node.
1390 1374 3 |
1391 1375 3 nml$q_entbfdsc [0] = nml$k_entbuflen;
1392 1376 3 nml$q_entbfdsc [1] = nml$t_entbuffer;
1393 1377 3 nml$getrecowner (node_rec_data,
1394 1378 3 nml$c_loopnode,
1395 1379 3 nml$q_entbfdsc,
1396 1380 3 nml$q_entbfdsc [0]);
1397 1381 3 nml$ab_msgblock [msb$a_entity] = nml$q_entbfdsc; ! Add entity descriptor pointer
1398 1382 3 nml$ab_msgblock [msb$l_flags] = msb$m_det_fld OR msb$m_entd_fld;
1399 1383 3 nml$ab_msgblock [msb$b_code] = nma$c_sts_pva;
1400 1384 3 nml$ab_msgblock [msb$w_detail] = nma$c_pcho_nli;
1401 1385 3 RETURN nml$sts_pva
1402 1386 3 END;
1403 1387 3 END;
1404 1388 3
1405 1389 3 The circuit is not already DEFINED for some other loopnode. Add it to
1406 1390 3 the node's permanent database record.
1407 1391 3
1408 1392 3 status = nml$defparam (.sem_list,
1409 1393 3 .bufdsc,
1410 1394 3 .length,
1411 1395 3 .addr,
```


: 1412
: 1413
: 14141396 2
1397 2 RETURN .status
1398 1 END; .rtnDsc);

! End of NML\$DEFNODNLI

				00FC	00000	.ENTRY	NML\$DEFNODNLI, Save R2,R3,R4,R5,R6,R7	1283	
		57	00000000G	00	9E	00002	MOVAB	NML\$SEARCHFLD, R7	
		56	00000000'	00	9E	00009	MOVAB	NML\$Q_ENTBFDSC, R6	
		55	00000000G	00	9E	00010	MOVAB	NML\$AB_MSGBLOCK, R5	
		5E	FBE4	CE	9E	00017	MOVAB	-1052(SP), SP	
				7E	D4	0001C	CLRL	FLDADR	1329
				5E	DD	0001E	PUSHL	SP	1330
			08	AE	9F	00020	PUSHAB	FLDSIZE	
		7E	01F6	8F	3C	00023	MOVZWL	#502, -(SP)	
			14	AC	DD	00028	PUSHL	RTNDSC	
		67		04	FB	0002B	CALLS	#4, NML\$SEARCHFLD	
		11		50	E9	0002E	BLBC	R0, 1\$	
		65		02	DD	00031	MOVL	#2, NML\$AB_MSGBLOCK	1339
04		A5		16	8E	00034	MNEGB	#22, NML\$AB_MSGBLOCK+4	1340
08		A5	01F5	8F	B0	00038	MOVW	#501, NML\$AB_MSGBLOCK+8	1341
		50		2C	CE	0003E	MNEGL	#44, R0	1342
				04	00	0041	RET		
			F8	AD	7C	00042	CLRQ	CIRCUIT_DSC	1345
			FC	AD	9F	00045	PUSHAB	CIRCUIT_DSC+4	1350
			F8	AD	9F	00048	PUSHAB	CIRCUIT_DSC	1349
		7E	01F5	8F	3C	0004B	MOVZWL	#501, -7SP)	1347
			14	AC	DD	00050	PUSHL	RTNDSC	
		67		04	FB	00053	CALLS	#4, NML\$SEARCHFLD	
		54		50	DD	00056	MOVL	R0, STATUS	
OC	AC		OC	54	E9	00059	BLBC	STATUS, 2\$	1356
			BD	F8	AD	2D	CMPCS	CIRCUIT_DSC, @CIRCUIT_DSC+4, #0, LENGTH, -	1357
				10	BC	00064		@ADDR	
				5A	13	00066	BEQL	3\$	
		10	AE	0400	8F	3C	MOVZWL	#1024, NODE_REC_DSC	1364
		14	AE	18	AE	9E	MOVAB	NODE_REC_BUF, NODE_REC_DSC+4	1365
		OC	AE	18	AE	9E	MOVAB	NODE_REC_BUF, NODE_REC_DATA+4	1366
				08	AE	9F	PUSHAB	NODE_REC_DATA	1367
				14	AE	9F	PUSHAB	NODE_REC_DSC	
			OC	AC	9F	0007E	PUSHAB	LENGTH	
		00000000G	00	03	FB	00081	CALLS	#3, NML\$READ_LOOPNODE	
			54	50	DD	00088	MOVL	R0, STATUS	
		0001827A	8F	54	D1	0008B	CMPL	STATUS, #9893B	1370
				2E	13	00092	BEQL	3\$	
			66	40	8F	9A	MOVZBL	#64, NML\$Q_ENTBFDSC	1375
		04	A6	C0	A6	9E	MOVAB	NML\$T_ENTBOFFER, NML\$Q_ENTBFDSC+4	1376
				56	DD	0009D	PUSHL	R6	1380
				56	DD	0009F	PUSHL	R6	1377
				05	DD	000A1	PUSHL	#5	
			14	AE	9F	000A3	PUSHAB	NODE_REC_DATA	
		00000000G	00	04	FB	000A6	CALLS	#4, NML\$GETREOWNER	
			14	A5	66	9E	MOVAB	NML\$Q_ENTBFDSC, NML\$AB_MSGBLOCK+20	1381
			65	12	DD	000B1	MOVL	#18, NML\$AB_MSGBLOCK	1382
			04	A5	10	8E	MNEGB	#16, NML\$AB_MSGBLOCK+4	1383
			08	A5	01F5	8F	MOVW	#501, NML\$AB_MSGBLOCK+8	1384


```
1416 1399 1 XSBTTL 'NML$DEFOBJNUM Add object number parameter'
1417 1400 1 GLOBAL ROUTINE NML$DEFOBJNUM (SEM_LIST, BUFDSC, LENGTH, ADDR, RTNDSC)=
1418 1401 1
1419 1402 1 ++
1420 1403 1 FUNCTIONAL DESCRIPTION:
1421 1404 1
1422 1405 1     This routine adds the object number parameter to the permanent
1423 1406 1     data base record if it is unique.
1424 1407 1
1425 1408 1 FORMAL PARAMETERS:
1426 1409 1
1427 1410 1     SEM_LIST      Parameter semantic table entry address.
1428 1411 1     BUFSIZE       Permanent database record maximum size.
1429 1412 1     LENGTH        Length of parameter to insert in record.
1430 1413 1     ADDR          Address of parameter to insert in record.
1431 1414 1     RTNDSC        Permanent database record buffer descriptor address.
1432 1415 1
1433 1416 1 IMPLICIT INPUTS:
1434 1417 1
1435 1418 1     It is assumed that the permanent data base file is already open.
1436 1419 1
1437 1420 1 IMPLICIT OUTPUTS:
1438 1421 1
1439 1422 1     The parameter is added to the record.
1440 1423 1
1441 1424 1 ROUTINE VALUE:
1442 1425 1 COMPLETION CODES:
1443 1426 1
1444 1427 1     Always returns success (NML$STS_SUC).
1445 1428 1
1446 1429 1 SIDE EFFECTS:
1447 1430 1
1448 1431 1     NONE
1449 1432 1
1450 1433 1 --
1451 1434 1 BEGIN
1452 1435 2
1453 1436 2 MAP
1454 1437 2 SEM_LIST : REF BBLOCK;
1455 1438 2
1456 1439 2 LOCAL
1457 1440 2 DUMDSC : DESCRIPTOR,
1458 1441 2 FLDADR,
1459 1442 2 FLDSIZE,
1460 1443 2 KEY : WORD,
1461 1444 2 STATUS;
1462 1445 2
1463 1446 2
1464 1447 2 FLDADR = 0;
1465 1448 2 FLDSIZE = 0;
1466 1449 2 STATUS = NMASSEARCHFLD (.RTNDSC,
1467 1450 2 NMASC PCOB_NUM,
1468 1451 2 FLDSIZE,
1469 1452 2 FLDADR);
1470 1453 2
1471 1454 2
1472 1455 2 ! If no object number is already defined or the object number is
```

```
1473 1456 2      | changed by the command, and
1474 1457 2      | the object number is not zero (duplicate objects numbered 0 are allowed),
1475 1458 2      | make sure that the new object number is not already in the
1476 1459 2      | permanent data base.
1477 1460 2      |
1478 1461 2      | IF (NOT .STATUS
1479 1462 2      |     OR (.STATUS AND CH$NEQ (.FLDSIZE, .FLDADR, .LENGTH, .ADDR)))
1480 1463 2      | AND CH$NEQ (.LENGTH, UPLIT(0), .LENGTH, .ADDR)
1481 1464 2      | THEN
1482 1465 2      |     BEGIN
1483 1466 2      |         KEY = 0;
1484 1467 2      |         IF NM$MATCHREC (NM$C_OPN_OBJ,
1485 1468 2      |                         NML$Q_PRMDSC,
1486 1469 2      |                         KEY,
1487 1470 2      |                         NM$C_PCOB_NUM,
1488 1471 2      |                         .LENGTH,
1489 1472 2      |                         .ADDR,
1490 1473 2      |                         DUMDSC)
1491 1474 2      |     THEN
1492 1475 2      |         BEGIN
1493 1476 4      |         Object number is not unique.
1494 1477 4      |
1495 1478 4      |         NML$AB_MSGBLOCK [MSB$L_FLAGS] = MSB$M_DET_FLD;
1496 1479 4      |         NML$AB_MSGBLOCK [MSB$B_CODE] = NM$C_STS_PVA;
1497 1480 4      |         NML$AB_MSGBLOCK [MSB$W_DETAIL] = NM$C_PCOB_NUM;
1498 1481 4      |
1499 1482 4      |         RETURN NML$STS_PVA
1500 1483 4      |
1501 1484 4      |     END;
1502 1485 4      | END;
1503 1486 3      |
1504 1487 2      | STATUS = NML$DEFPARAM (.SEM_LIST,
1505 1488 2      |                      .BUFDSC,
1506 1489 2      |                      .LENGTH,
1507 1490 2      |                      .ADDR,
1508 1491 2      |                      .RTNDSC);
1509 1492 2      |
1510 1493 2      | RETURN .STATUS
1511 1494 2      |
1512 1495 2      | END;
1513 1496 2      |
1514 1497 1      |
```

! End of NML\$DEFOBJNUM

.PSECT \$SPLITS, NOWRT, NOEXE, 2

00000000 00031 .BLKB 3
00034 P.AAG: .LONG 0

.PSECT \$CODE\$, NOWRT, 2

55 00000000G 003C 00000
5E 00 9E 00002
10 C2 00009.ENTRY NML\$DEFOBJNUM, Save R2, R3, R4, R5
MOVAB NML\$AB_MSGBLOCK, R5
SUBL2 #16, SP: 1400
:
:

				7E	D4	0000C		CLRL	FLDADR	1447
			04	AE	D4	0000E		CLRL	FLDSIZE	1448
				5E	DD	00011		PUSHL	SP	1449
			08	AE	9F	00013		PUSHAB	FLDSIZE	
		7E	0201	8F	3C	00016		MOVZWL	#513, -(SP)	
			14	AC	DD	0001B		PUSHL	RTND\$C	
		00000000G	00	04	FB	0001E		CALLS	#4, NMASSEARCHFLD	
			54	50	D0	00025		MOVL	R0, STATUS	
			OC	54	E9	00028		BLBC	STATUS, 1\$	1461
OC	AC			04	AE	2D	0002B	CMPC5	FLDSIZE, @FLDADR, #0, LENGTH, @ADDR	1462
				10	BC		00033			
				41	13		00035	BEQL	2\$	
		10	BC	00000000'	00	OC	29	CMPC3	LENGTH, P.AAG, @ADDR	1463
						35	13	BEQL	2\$	
				08	AE	B4	00043	CLRW	KEY	1467
				OC	AE	9F	00046	PUSHAB	DUMDSC	1468
			7E	OC	AC	7D	00049	MOVQ	LENGTH, -(SP)	1472
			7E	0201	8F	3C	0004D	MOVZWL	#513, -(SP)	1468
				18	AE	9F	00052	PUSHAB	KEY	
				00000000'	00	9F	00055	PUSHAB	NML\$Q_PRMDSC	
					03	DD	0005B	PUSHL	#3	
		00000000G	00	07	FB	0005D		CALLS	#7, NMASMATCHREC	
			11	50	E9	00064		BLBC	R0, 2\$	1480
			65	02	D0	00067		MOVL	#2, NML\$AB_MSGBLOCK	1481
		04	A5	10	8E	0006A		MNEGB	#16, NML\$AB_MSGBLOCK+4	1482
		08	A5	0201	8F	B0	0006E	MOVW	#513, NML\$AB_MSGBLOCK+8	1484
			50	20	CE	00074		MNEGL	#32, R0	
					04		00077	RET		
			7E	10	AC	7D	00078	MOVQ	ADDR, -(SP)	1492
			7E	08	AC	7D	0007C	MOVQ	BUFD\$C, -(SP)	1490
				04	AC	DD	00080	PUSHL	SEM_LIST	1489
		FC62	CF	05	FB	00083		CALLS	#5, NML\$DEFPARAM	
			54	50	D0	00088		MOVL	R0, STATUS	
					04		0008B	RET		1497

; Routine Size: 140 bytes, Routine Base: \$CODE\$ + 0626

```
1516 1498 1 %SBTTL 'NML$PURPARAM Delete parameter'
1517 1499 1 GLOBAL ROUTINE NML$PURPARAM (RTNDSC, SEM_LIST)=
1518 1500 1
1519 1501 1 ++
1520 1502 1 FUNCTIONAL DESCRIPTION:
1521 1503 1
1522 1504 1 This routine removes a parameter from the permanent data base record.
1523 1505 1
1524 1506 1 FORMAL PARAMETERS:
1525 1507 1
1526 1508 1 SEM_LIST Parameter semantic table entry address.
1527 1509 1 RTNDSC Record buffer descriptor address.
1528 1510 1
1529 1511 1 IMPLICIT INPUTS:
1530 1512 1
1531 1513 1 It is assumed that the permanent data base file is already open.
1532 1514 1
1533 1515 1 IMPLICIT OUTPUTS:
1534 1516 1
1535 1517 1 The parameter has been removed from the record.
1536 1518 1
1537 1519 1 ROUTINE VALUE:
1538 1520 1 COMPLETION CODES:
1539 1521 1
1540 1522 1 Always returns success (NML$STS_SUC).
1541 1523 1
1542 1524 1 SIDE EFFECTS:
1543 1525 1
1544 1526 1 NONE
1545 1527 1
1546 1528 1 --
1547 1529 1
1548 1530 2 BEGIN
1549 1531 2
1550 1532 2 MAP
1551 1533 2 SEM_LIST : REF BBLOCK;
1552 1534 2
1553 1535 2 NML$DELETEFLD (.RTNDSC,
1554 1536 2 .SEM_LIST [PST$W_DATAID]);
1555 1537 2
1556 1538 2 RETURN NML$STS_SUC
1557 1539 2
1558 1540 1 END;
```

! End of NML\$PURPARAM

```
0000 00000
7E 08 BC 3C 00002
04 AC DD 00006
00000000G 00 02 FB 00009
50 01 D0 00010
04 00013
```

```
.ENTRY NML$PURPARAM, Save nothing
MOVZWL @SEM_LIST, -(SP)
PUSHL RTNDSC
CALLS #2, NML$DELETEFLD
MOVL #1, R0
RET
```

```
: 1499
: 1536
: 1535
:
: 1538
: 1540
```

; Routine Size: 20 bytes, Routine Base: \$CODE\$ + 06B2

NMLSLSPRM
V04-000

NML special parameter handling routines
NMLSPURPARAM Delete parameter

H 6
16-Sep-1984 00:16:56
14-Sep-1984 12:50:09

VAX-11 Bliss-32 V4.0-742
[NML.SRC]NMLLISPRM.B32;1

Page 49
(19)

NML
V04

```
1560 1541 1 %SBTTL 'NML$PURNODNNA Delete node name parameter'
1561 1542 1 GLOBAL ROUTINE NML$PURNODNNA (RTNDSC, SEM_LIST)=
1562 1543 1
1563 1544 1 ++
1564 1545 1 FUNCTIONAL DESCRIPTION:
1565 1546 1 This routine removes the node name parameter from the permanent
1566 1547 1 data base record if it is not required. It is required in the case
1567 1548 1 of a loop node.
1568 1549 1
1569 1550 1 FORMAL PARAMETERS:
1570 1551 1 RTNDSC Data buffer descriptor address.
1571 1552 1 SEM_LIST Parameter semantic table entry address.
1572 1553 1
1573 1554 1 IMPLICIT INPUTS:
1574 1555 1 It is assumed that the permanent data base file is already open.
1575 1556 1
1576 1557 1 IMPLICIT OUTPUTS:
1577 1558 1 NONE
1578 1559 1
1579 1560 1 ROUTINE VALUE:
1580 1561 1 COMPLETION CODES:
1581 1562 1 Error is returned if the parameter cannot be removed.
1582 1563 1
1583 1564 1 SIDE EFFECTS:
1584 1565 1 NONE
1585 1566 1
1586 1567 1 --
1587 1568 1
1588 1569 2 BEGIN
1589 1570 2
1590 1571 2 MAP
1591 1572 2 SEM_LIST : REF BBLOCK;
1592 1573 2
1593 1574 2 LOCAL
1594 1575 2 FLDADR,
1595 1576 2 FLDSIZE;
1596 1577 2
1597 1578 2 FLDADR = 0;
1598 1579 2 FLDSIZE = 0;
1599 1580 2 IF NMA$SEARCHFLD (.RTNDSC,
1600 1581 2 NMA$C_PCNO_NLI,
1601 1582 2 FLDSIZE,
1602 1583 2 FLDADR)
1603 1584 2 THEN
1604 1585 2 BEGIN
1605 1586 2
1606 1587 2 Node has circuit (is a loopnode) so name cannot be deleted.
1607 1588 2
1608 1589 2 NML$AB_MSGBLOCK [MSB$S_FLAGS] = MSB$M_DET_FLD;
1609 1590 2 NML$AB_MSGBLOCK [MSB$B_CODE] = NMA$C_STS_PNA;
1610 1591 2 NML$AB_MSGBLOCK [MSB$W_DETAIL] = NMA$C_PCNO_NNA;
1611 1592 2
1612 1593 2 RETURN NML$STS_PNA
1613 1594 2
1614 1595 2 END
1615 1596 2 ELSE
1616 1597 2 NMA$DELETEFLD (.RTNDSC, .SEM_LIST [PST$W_DATAID]);
```


: 1617
: 1618
: 1619
: 16201598 2
1599 2 RETURN NML\$_STS_SUC
1600 2
1601 1 END;

! End of NML\$PURNODNNA

			0004 00000	.ENTRY	NML\$PURNODNNA, Save R2	: 1542
52	00000000G	00	9E 00002	MOVAB	NML\$AB_MSGBLOCK, R2	
5E		04	C2 00009	SUBL2	#4, SP	
		7E	D4 0000C	CLRL	FLDADR	: 1578
		04	AE D4 0000E	CLRL	FLDSIZE	: 1579
		5E	DD 00011	PUSHL	SP	: 1580
		08	AE 9F 00013	PUSHAB	FLDSIZE	
7E	01F5	8F	3C 00016	MOVZWL	#501, -(SP)	
	04	AC	DD 0001B	PUSHL	RTND\$C	
00000000G	00	04	FB 0001E	CALLS	#4, NMA\$SEARCHFLD	
	11	50	E9 00025	BLBC	R0, 1\$	
	62	02	D0 00028	MOVL	#2, NML\$AB_MSGBLOCK	: 1589
04	A2	16	8E 0002B	MNEGB	#22, NML\$AB_MSGBLOCK+4	: 1590
08	A2	8F	B0 0002F	MOVW	#500, NML\$AB_MSGBLOCK+8	: 1591
	50	2C	CE 00035	MNEGL	#44, R0	: 1593
		04	00038	RET		
7E	08	BC	3C 00039	MOVZWL	@SEM_LIST, -(SP)	: 1597
	04	AC	DD 0003D	PUSHL	RTND\$C	
00000000G	00	02	FB 00040	CALLS	#2, NMA\$DELETEFLD	
	50	01	D0 00047	MOVL	#1, R0	: 1599
		04	0004A	RET		: 1601

: Routine Size: 75 bytes, Routine Base: \$CODE\$ + 06C6

NML\$LISPRM
VO4-000

NML special parameter handling routines
NML\$PURNODNNA Delete node name parameter

K 6
16-Sep-1984 00:16:56
14-Sep-1984 12:50:09

VAX-11 Bliss-32 V4.0-742
[NML.SRC]NMLLISPRM.B32;1

Page 52
(21)

: 1622
: 1623
: 1624
1602 1 END
1603 1
1604 0 ELUDOM

PSECT SUMMARY

Name	Bytes	Attributes
\$OWNS	334	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$PLITS	56	NOVEC, NOWRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODES	1809	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
\$255\$DUA28:[NML.OBJ]NMLLIB.L32;1	341	42	12	27	00:00.1
\$255\$DUA28:[SHRLIB]NMLIBRY.L32;1	887	21	2	47	00:00.2
\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	4	0	581	00:02.2

COMMAND QUALIFIERS

: BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:NMLLISPRM/OBJ=OBJ\$:NMLLISPRM MSRC\$:NMLLISPRM/UPDATE=(ENH\$:NMLLISPRM)

: Size: 1809 code + 390 data bytes
: Run Time: 00:34.6
: Elapsed Time: 01:30.8
: Lines/CPU Min: 2781
: Lexemes/CPU-Min: 13283
: Memory Used: 131 pages
: Compilation Complete

0284 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

